Get-GPTrashFire

Identifying and Abusing Vulnerable Configurations in MS AD Group Policy

I’m this guy.

- I’m Mike Loss
- @mikeloss on Twitter 
- @l0ss on Slack
- I do breaking of things.
- I’m not really that guy.
- That’s Brian Blessed.

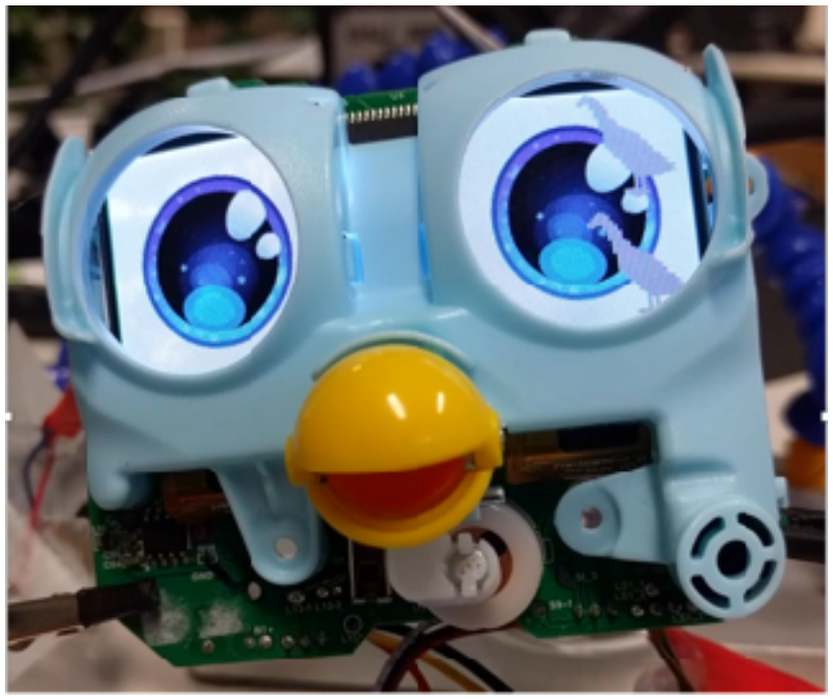
- Thanks
- my name’s Mike Loss
- I’m lucky enough to get to break other people’s things for a living.

I work for one of these guys.

IX	ISK
	

I work for a company called Asterisk Information Security in Perth.

I was mean to this guy.



You may remember me from last year’s BSidesCBR
where my pal Clinton and I made a Furby say a swear.

“You could do a whole talk about owning stuff using screwups in Group Policy.”

–Me, Self-owning at WAHCKon}`OxV)] - 2017

Less of you may also know me from a talk I did at WAHCKon about Active Directory privilege escalation.
In that talk I said Group Policy probably deserves a talk all of its own.
This is that talk.
No Trump pictures this time, I promise.

I’m gonna try to answer these questions:

- What is Group Policy?
- Why should I, a cool hacker person, give a crap?
- How do I look at it?
- What fun things can I do with it?
- Is there a script? I’m a hacker, and hackers like scripts.

- I’m gonna talk about Group Policy.
- We’re gonna cover what it is,
- why I think you might care,
- a few different ways of looking at it,
- the fun things you might find and do with it,
- and a script.
- (more notes next slide)

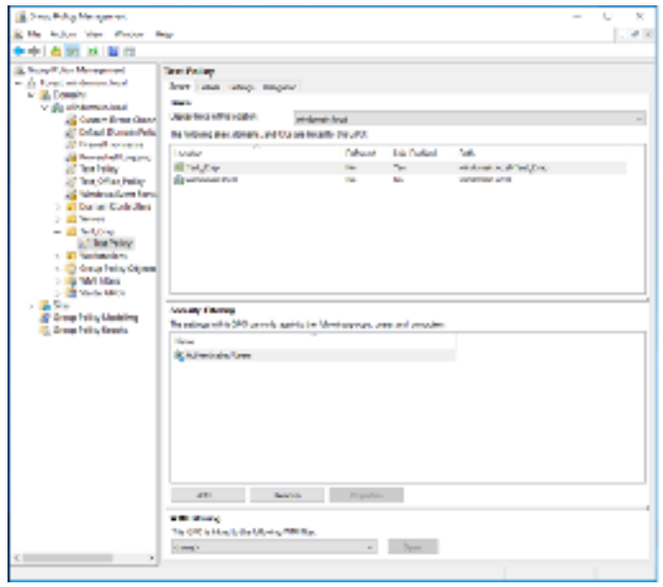
I’m gonna try to answer these questions:

- What is Group Policy?
- Why should I, a cool hacker person, give a crap?
- How do I look at it?
- What fun things can I do with it?
- Is there a script? I’m a hacker, and hackers like scripts.

I’m not
gonna be dropping any sick 0day,
gonna go into all that much detail because it’s not that kind of talk.
Much like with my last AD-related talk, I’m going to just talk about what you CAN do.
not much detail on how
when the time comes you can google that.

What is Group Policy?

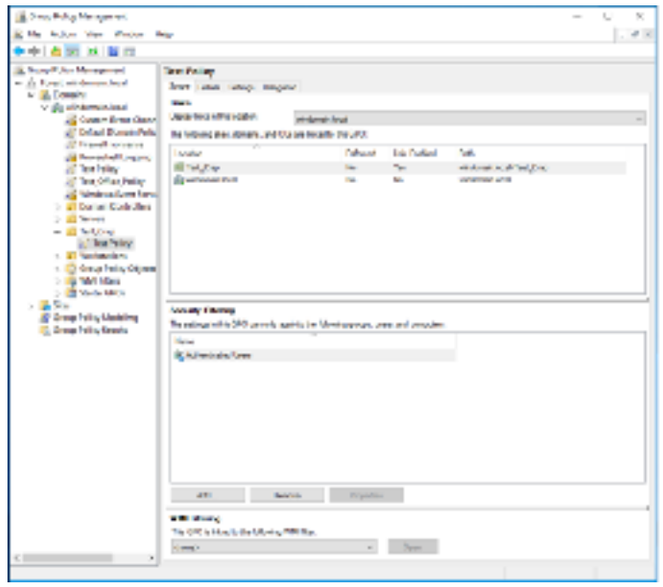
- Built into Microsoft AD
- Sort-of configuration management system
- Configure settings in Group Policy Objects (GPOs)
  - Assign GPOs to OUs (folders) in AD
  - Settings apply automatically to users and computers in that OU
  - Applied every 90 mins (with 30 minute jitter) - every 5 minutes on Domain Controllers



Component of Active Directory, acts as a sort-of configuration management system.
Things called Group Policy Objects, which each contain a bunch of settings.
GPOs get assigned to OUs or “Folders”
Settings get applied to all users or computers in that OU
At boot (for computers),
logon (for users), and then every 90 mins thereafter (with 30 minute jitter) or every 5 minutes for Domain Controllers.

What is Group Policy?

- Built into Microsoft AD
- Sort-of configuration management system
- Configure settings in Group Policy Objects (GPOs)
  - Assign GPOs to OUs (folders) in AD
  - Settings apply automatically to users and computers in that OU
  - Applied every 90 mins (with 30 minute jitter) - every 5 minutes on Domain Controllers



Common uses include:

- allowing that one control-freak sysadmin to force a bunch of pointless cosmetic changes on everyone in the company

AND

- applying horrible vulnerabilities in a consistent manner to every host in the domain.

If used well, it’s a decent-ish system for mass deploying configs to Windows hosts, especially given how old it is.

Why should I care?

Why should you care?
Because every time an admin configures a thing, that's an opportunity for an admin to make a mistake, and in doing so, make that thing vulnerable.

Why should I care?

Like any powerful tool, Group Policy provides an extraordinary number of opportunities for admins to make really horrible mistakes that tend to stick around forever.

And because (as we all know) all hackers are terrible people, we always like it when people make horrible mistakes.

Really horrible mistakes.

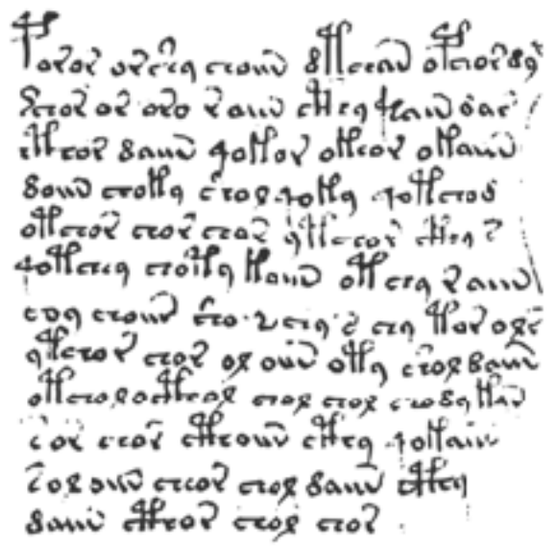
OK, maybe not always.

OK, I want to gawp at other people's horrible mistakes. How do I do that?

So one common mistake made by Windows admins is to assume they are the only ones who can read Group Policy.
One of the more straightforward rules of Group Policy is that for a policy object to apply to you, you HAVE to be able to read it.
As a result, by default, all domain users can read all GPOs.
This means that if you have any domain creds at all, you can probably read most if not all the GPOs in the domain.

Get-…pseudo-legible crap from SYSVOL?

`\\domain.tld\SYSVOL\domain.tld\Policies`



So if you want to, you can go straight to the source.
AD has a special file share called SYSVOL for storing (among other things) group policy.
At first this seems great, because in theory you can just dump the entire directory and read stuff from there, but then you realise that the GPO folders are named by UID instead of anything descriptive, and the files within range in legibility between ‘not too bad’ XML and ‘proprietary binary nightmare’ depending on the settings being defined. But the data is all there.
There’s also the GUI app - Group Policy Management Console or GPMC, but don’t use that, it's 2018…

Get-GPOReport

- If you're on a machine that is a domain member:

```
Get-GPOReport -All -ReportType Html -Path C:\temp\gporeport.html -Domain $domain
```

- If not, do this first:

```
C:\windows\system32\runas.exe /netonly /user:user@domain.tld powershell.exe
```

- If you want an XML report (you do) then do it with:

```
-ReportType XML
```

…use PowerShell.

So the Get-GPOReport cmdlet from Microsoft's Group Policy module is very handy.

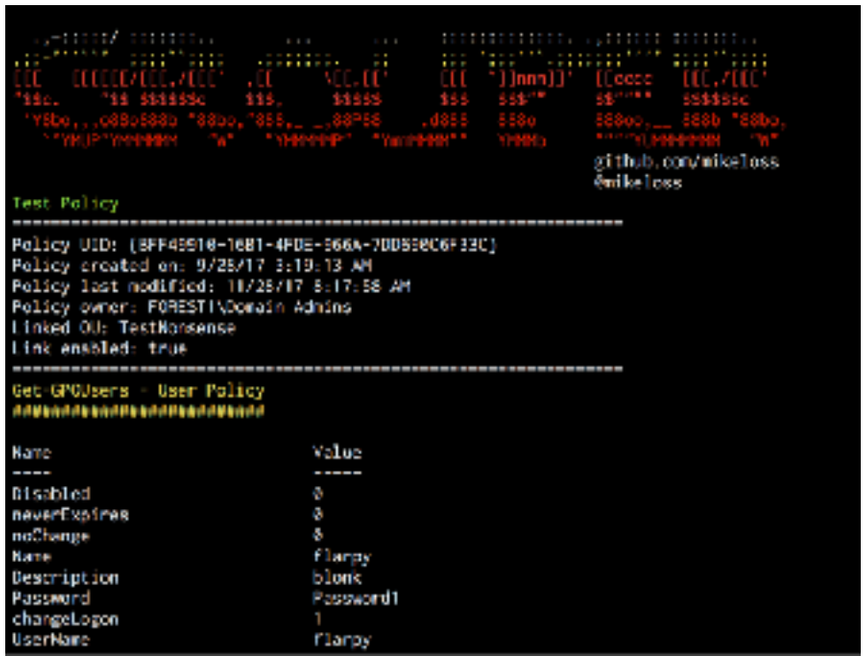
It can be very easily used to grab a full report of every GPO in the domain, in your choice of not-very-nice HTML, or equally unpleasant XML.

For those of you familiar with harmj0y's PowerView module, it has some very useful cmdlets for working with Group Policy, and it's particularly handy for figuring out which GPOs apply to which users and computers.

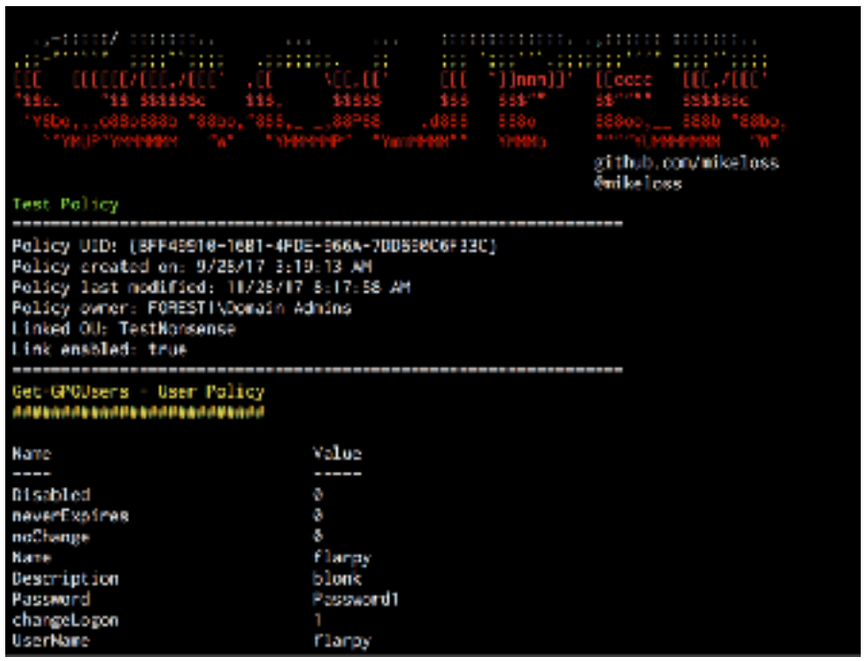
So now I have 165MB of barely-legible HTML that crashes every browser I try to view it in, and an enormous wad of XML with a structure that is inconsistent at best.

This is hard and gross and I'm angry and I want someone else to make it easier.

I spent a lot of time on engagements scrolling through HTML reports so big that browsers would crash when you tried to load them.
So if you have actually done this stuff before, I know what you're thinking:
This is not a good way of looking at this stuff.
Surely someone else has already solved this problem?
So I dredged through the internet looking for someone who had, and I drew a complete blank, so I had a swing at it.



I wrote 1800 lines of XML parsing nightmare in PowerShell because I am giant idiot.



I wrote 1800 lines of XML parsing nightmare in PowerShell because I am giant idiot.

Grouper is a thing I wrote to basically filter out all of the useless noise in the output of Get-GPOReport.
It’s pretty simple to use, you just tell it the path of the XML file,
Then (depending on the verbosity level you choose) it’ll show you all the settings that in some way relate to security, or just the settings that Grouper thinks have been screwed up badly enough to let you own stuff.
By default it doesn’t touch the network at all, it only parses the report XML, but there’s a switch that’ll turn on a bunch of checks to see if some settings are exploitable as the current user.

The Goods

So now I’m going to dig into the fun stuff and talk about each of the classes of security-related settings you can find in Group Policy, the ways they get screwed up, and how you can exploit them.

If you see this little guy appear in the corner…

The Goods

That’s the shell bell!
If you see the shell bell, it’s reasonably straightforward to use the thing we’re looking at to pop shell, if someone screwed up badly enough.

Get-GPOACLs

- GPOs have ACLs.
- Admins hand out excessive privilege.
- If you can edit a GPO, you own every user and computer it applies to.

```
TattCorp_Workstation_Policy
================================================================
Policy UID: {D7C7B23B-B33E-4172-9AED-F1A2AB521BB9}

Policy created on: 1/30/18 12:19:00 PM
Policy last modified: 1/30/18 12:22:49 PM
Policy owner: forest1\Domain Admins

Linked OU: Work
Link enabled: true
================================================================
GPO Permissions
###############
Trustee                         AD\People_With_Tribal_Tattoos...
Access                          Own stuff.
Type                            Allow
```

So, like many other things in Windowsland, GPOs are “securable objects”, meaning that they have ACLs.
Like all things in all operating systems, Admins will happily hand out excessive permissions if it will make people leave them alone.
If you get write permissions on a GPO, you can basically own every user or computer that it applies to. The New-GPOImmediateTask cmdlet from PowerView is the easiest way - it just creates a scheduled task that runs and then deletes itself.
By the way, I saw RastaMouse mention on twitter the other day that he’s working on a PowerShell module to abuse a bunch of this stuff “in ways other than New-GPOImmediateTask”. I haven’t seen any more detail since then but should be fun.

Get-GPORegKeys

- Manually defined registry entries, never know what you'll find here.
- VNC password
- AV disable password
- Autologon password

```
Get-GPORegKeys - User Policy
############################
Action                         U
Value                          asdgasdg
Key                            SOFTWARE
Hive                           HKEY_LOCAL_MACHINE
Name                           asdf
```

This is the most flexible class of settings, and it's where admins can set up any custom registry entries they see fit.
You'll find a bunch of fun stuff in here.
The most useful examples are things like VNC passwords, which are encrypted using a standard method and key, so you can easily decrypt those and have a nice time.
Another favourite is, Autologon credentials. These aren't stored encrypted, just right there in the registry in plain text. These will often be pretty rubbish stuff like low-privilege accounts for kiosk machines but sometimes you'll get something great like "the account used for the big TV in the network admins' area that shows all of their monitoring stuff and therefore had to have access to all the monitoring stuff and it was easiest to just make it a domain admin."
This category tends to be very noisy, especially in shops with very old-school admins.

Get-GPORegSettings

- All kinds of settings that are defined in the registry
- If you add extra policy templates (ADMX files) to Group Policy, this is where those policies show up, e.g. MS Office, Citrix, etc.

```
Get-GPORegSettings - Computer Policy
####################################
State                           Enabled
Setting Name                    Internet proxy servers for apps
Supported                       At least Windows Server 2012, Windows 8 or Windows RT
Explain                          This setting does not apply to desktop apps....
Category                        Network/Network Isolation
State                           Enabled
Name                            Domain Proxies
Value                           internal.proxy.tld


State                           Enabled
Setting Name                    Intranet proxy servers for  apps
Supported                       At least Windows Server 2012, Windows 8 or Windows RT
Explain                          This setting does not apply to desktop apps....
Category                        Network/Network Isolation
State                           Enabled
Name                            Type a proxy server IP address for the intranet
Value                           whee.proxy.secret
```

Despite the similar name, not the same thing - this category is full of Windows configuration options that are ALSO set in the registry, but group policy provides a nice GUI interface for admins to set them rather than having to define the keys manually. There's some fun potentially abusable bits in here, and unfortunately an enormous amount of very tedious crud.

One less-tedious thing is AlwaysInstallElevated. This is a fun option that grants Push-button privesc. Basically this says ‘all WMI installers run as admin’. msfvenom will output a wmi to abuse it easily, as will PowerUp.

This section will also see which servers are used for windows updates, maybe try out that nifty WSUS attack those french guys presented at defcon last year?

All your MS office settings live in here too. Mostly boring, but if you can write to the path where the default office templates live, you can do something like put some fun macros in them.

Get-GPOFWSetting

- Windows Firewall settings.
- Useful to know, not gonna get you shell on its own.

Windows Firewall settings:
This is a good example of where taking the time to read Group Policy settings isn't gonna get you a shell on its own, but can provide a lot of really useful information. For example, host based firewall settings will tell you what kind of services you might expect to see listening on machines, without having to do any nasty noisy port scanning.
It also helps substantially in answering questions like 'why isn't my reverse shell working?'.

Get-GPOUsers

- Creating and modifying and deleting local users.
- Good old fashioned GPP Passwords

```
Get-GPOUsers - Computer Policy
##############################
Disabled                       0
neverExpires                   0
noChange                       0
Name                           scrooge
Description                    quack quack
Password                       schedtaskpass
changeLogon                    0
UserName                       scrooge
```

Get-GPOUsers
This is where our old friends Group Policy Preferences passwords live. These were such a problem that Microsoft pushed out a patch to stop admins creating new ones. Basically if an admin used this feature to set local creds on a host, you can just read them. They’re encrypted, but using a single universal key that is published on TechNet.
Grouper will decrypt these creds for you, although for completeness it might be worth using PowerSploit’s “Get-GPPPassword” too, because there’s an edge-case I ran into a couple weeks back, where a replication error a long time ago can create old broken policy folders that stick around but can’t be seen in the GPOReport that Grouper reads.
This section can also include a bunch of handy information like 'they changed the "administrator" account to be named "sysadmin" ' or 'they created a local service account on this machine with a note saying 'username = password' in the description field.’

Get-GPOGroups

- Changes to local groups
  - Adding/removing local and domain users and groups to them
  - Creating new ones

```
Get-GPOGroups - Computer Policy
###############################
Group Name                          Backup Operators
Name                                Backup Operators
Action                              ADD
Name                                Domain Users
```

This is where you can see users getting added to or removed from local groups, most often the local administrators and remote desktop users groups.
Don't forget though, that there are good times to be had from groups like 'backup operators' which is Microsoft-speak for 'read-write access to the whole FS as long as you use the backup API to do it’.
This policy setting goes a long way to helping find those machines that your account has RDP access to, but which tools like BloodHound won’t point out because you don’t have full admin access.

Get-GPOFileUpdate

- Usually used to copy a file from file share > local FS
- If it’s an executable, and you can edit it…
- Config files often have creds, e.g. SQL conn. strings

```
Get-GPOFileUpdate - User Policy
###############################
Action                          U
targetPath                      C:\Users\Scrooge\Desktop\fix_things.ps1
Name                            fix_things.ps1
fromPath                        \\badly\secured\network\share\hack_things.ps1

Get-GPOFileUpdate - Computer Policy
###################################
Action                          U
targetPath                      C:\Users\l0ss\Desktop\eyebleachPls.ps1
Name                            thing3.bat
fromPath                        \\ive.seen.things\I\cant\unsee\
```

This is for modifying, replacing, moving, adding files to a windows system. Commonly used for things like updating local config files with cleartext credentials in them, and it will often be used to "Install" simple in-house developed executables or scripts (from a network drive with weak file permissions no doubt) to paths on the local filesystem.
In -online mode, Grouper will return the file permissions for the source file and will also check if the current user can modify it.
Depending on the type of file being placed, you might need to get creative to do anything with this, but if it’s just a script or an exe, and it actually gets run, sub it out for something fun, have a real nice time.

Get-GPOMSIInstallation

- Does what it says on the tin.
- Installs an application from an MSI file, usually on a fileshare.
- If you can edit the MSI file… you get the picture.

```
Get-GPOMSIInstallation - Computer Policy
########################################
Name                            Goofy Font Pack
Path                            \\windomain.local\SYSVOL\windomain.local\scripts\MSI\Fonts_Not_Malware.msi
```

This one… does what it says on the tin. Loads a Windows MSI installer file from somewhere on the network and installs it on the host.
Again, Grouper in -online mode will tell you who can modify the source file. If that who is you, then… ding!
This is the one I see most often in the wild so definitely check it out.

Get-GPOSchedTasks

- Creates, modifies, and deletes scheduled tasks on the target host.
- Definition can include creds
- If the thing being run is on a network share and you can edit it…
- Don’t forget to look at args!

```
Get-GPOSchedTasks - Computer Policy
###################################
args                            -doesCalcHaveArgs
Action                          U
runAs                           Donald
Password                        schedtaskpass
Name                            schedtask
appName                         calc.exe
startMinutes                    0
startHour                       22
type                            DAILY
```

- Sets up scheduled tasks on the host. There’s a bunch of different fun ways to screw these up.
- they often run a binary or a script from a file share (and yes grouper will check if you can modify it)
- will often be configured to run as a service account and then the creds are stored in the policy object using the same encryption method used when adding local users.
- Admins who think that only they can read GPOs will think they’re being secure by not hard-coding creds in a script, but then they pass the creds as arguments defined in the GPO.

Get-GPOFolderRedirection

- Redirects folders in a user’s homedir.
- Exploitation difficulty varies from ‘trivial’ to ‘time to get creative’.

```
Get-GPOFolderRedirection - All Policy
#####################################
ID                                  {FDD39AD0-238F-46AF-AD84-8C85480359C7}
Target Group                        FOREST1\lowprivusers
DestPath                            C:\temp\%USERNAME%\Documents
Target SID                          s-1-5-21-888228836-1552922583-4278570199-1110
```

This nifty feature lets admins reconfigure users' home directories so that their Desktop or Documents folders are stored on a network file share with badly configured permissions. There’s plenty of ways to get cute and creative with this one.

- Replace a desktop shortcut with one that points at a malicious exe,
- or drop a shortcut file with the icon set to point at a machine running responder,
- or just steal passwords.txt from their desktop,
- or jam a nasty macro in one of their office docs.
- So many opportunities to get creative.

Get-GPOSecurityOption

- Misc security guff, surprisingly not that exciting
- Lots of stuff that could be useful in the right circumstances

```
Get-GPOSecurityOptions - All Policy
###################################
Name                        Network access: Remotely accessible registry paths
KeyName                     MACHINE\System\CurrentControlSet\Control\SecurePipeServers\Winreg\AllowedExactPaths\Machine
Path/Pipe1                  Software\Microsoft\Windows NT\CurrentVersion
Path/Pipe4                  System\CurrentControlSet\Control\ProductOptions
Path/Pipe7                  System\CurrentControlSet\Control\Server Applications


Name                        Network access: Remotely accessible registry paths and sub-paths
KeyName                     MACHINE\System\CurrentControlSet\Control\SecurePipeServers\Winreg\AllowedPaths\Machine
Path/Pipe4                  Software\Microsoft\OLAP Server
Path/Pipe2                  System\CurrentControlSet\Control\Print\Printers
Path/Pipe3                  System\CurrentControlSet\Control\Terminal Server\UserConfig
Path/Pipe6                  System\CurrentControlSet\Control\Terminal Server
Path/Pipe8                  System\CurrentControlSet\Control\Terminal Server\DefaultUserConfiguration
Path/Pipe5                  System\CurrentControlSet\Control\ContentIndex
Path/Pipe6                  Software\Microsoft\Windows NT\CurrentVersion\Print
Path/Pipe1                  System\CurrentControlSet\Services\Eventlog
Path/Pipe7                  Software\Microsoft\Windows NT\CurrentVersion\Windows
Path/Pipe10                 System\CurrentControlSet\Services\Sysmonlog
Path/Pipe9                  Software\Microsoft\Windows NT\CurrentVersion\Perflib
```

This is kind of the 'Misc' category for security settings - you'll find some interesting esoterica in here like which named pipes can be accessed anonymously, or which registry paths can be accessed remotely, whether it's possible for an account with a blank password to log in to the host, whether or not the host stores passwords as old-school LM hashes, that kind of thing.
There are things that an admin could screw up super badly in here, and they’re worth exploring if you see them, but the stuff that will actually let you pwn stuff is very unlikely to be messed up because admins don’t often have reason to mess with it.

Get-GPOAccountSetting

- Password policy
- Account lockout policy
- Cleartext password storage

```
Get-GPOAccountSettings - All Policy
###################################
Type                                Password
SettingBoolean                      true
Name                                ClearTextPassword
```

This is where you'll find stuff like password complexity policy, account lockout policy, whether passwords can be stored in the clear, that kind of stuff.

Very useful stuff to know if you’re trying to password spray or something, but there’s nothing insanely sexy in here.

Get-GPOScripts

- Startup and shutdown scripts.
- If you can edit them, you own the user/computer they apply to.
- Sometimes they have hard-coded creds in them. Delightful!

```
Get-GPOScripts - Computer Policy
################################
Parameters                      -dont googleBadTattoos
Command                         \\i.regret.the\tattoo\theme.bat
Type                            Startup
```

This is where you can find startup/shutdown/logon/logoff scripts.
If you're really lucky you'll find fun creds hard-coded in them.
Most of them will be stored in SYSVOL and their file permissions won’t let you change them, but some will be on a random file server where an admin granted "Full Control" to the "Everyone" group because they were hung over or something. Once again, Grouper in -Online mode will tell you if you can modify the script in question.

Once again, you’ll also sometimes find creds being passed as arguments in these entries, in the name of ‘security’.

Get-GPONetworkShares

- Creates, modifies, or deletes file shares.

```
Get-GPONetworkShares - All Policy
#################################
Action                          U
Path                            C:\temp\backups\sql\
Name                            OverShare
PropName                        OverShare
```

Group Policy can also be used to set up network shares on hosts - definitely worth checking out for that 1 in 20 chance that all the devs keep backups of the prod MSSQL databases on file-shares on their workstations because... reasons?

Get-GPOIniFiles

- Sets application configs in .INI files.
- Rarely used, but could contain some fun stuff.

Get-GPOIniFiles

Ini files are used by a buttload of Windows apps for storing configuration options. I've never actually seen group policy used to push them out in the wild, but given that INI files sometimes contain stuff like DB connection strings, credentials, crypto keys, etc - it's definitely worth having a squiz at.

Get-GPOEnvVars

- I have never seen an admin use this.
- I can imagine it being used in insane ways, so I’m including it.

```
Get-GPOEnvVars - User Policy
############################
Action                         U
Value                          asdf
Status                         moar_env_vars_here = asdf
Name                           moar_env_vars_here
```

Get-GPOEnvVars
Like it says on the tin - allows admins to set environment variables.
I’ve never seen this used either, but I’ve seen setups on Linux servers that do unhinged shit like set AWS tokens as environment variables, so you don’t want to miss it when the day comes.

Get-GPOShortcuts

- Pushes .lnk files
- Good for finding commonly used and critical apps on file shares.
- If you can modify the target…

```
Get-GPOShortcuts - Computer Policy
##################################
startIn                          C:\temp\doodles
Name                             shortcutname
targetType                       FILESYSTEM
targetPath                       C:\temp\doodles
arguments                        -argument1
Action                           U
shortcutPath                     %DesktopDir%\shortcutname
Status                           shortcutname
```

Get-GPOShortcuts
Deploys .lnk files. These are useful for two reasons, the first being that admins love to set up the helpdesk with web bookmarks for all those awful internal web apps that do insanely powerful things like reset user passwords or provision MFA tokens, so you can find those and make a real mess.
The second is that (you guessed it) they often point to binaries and scripts on weakly protected file shares.
We tamper, we have nice time.

Get-GPOUserRights

- Do all kinds of fun stuff:
  - SeRemoteInteractiveLogonRight = RDP Access
  - SeTcbPrivilege = “Act as part of the operating system”
  - SeMachineAccountPrivilege = Add machines to domain
  - SeBackupPrivilege = Read any file via backup API
  - AND MANY MORE!!!

```
Get-GPOUserRights - All Policy
##############################
Right                          SeRemoteInteractiveLogonRight
Members                        Actually_Literally_Everyone
```

Get-GPOUserRights
Windows separates out the concept of 'rights' from the concept of 'permissions' and it can be super confusing for people more familiar with Linux.
Some of these 'rights' are pretty mundane, but some are quite insanely powerful and let you do things like RDP to the host, take ownership of any file, or debug any process, or “act as part of the operating system”. All the best ones boil down to ‘let you mimikatz’.
Anyway, Grouper sorts the wheat from the chaff for you in this category - it only shows you the rights that have meaningful security implications.
The most common one of these that you’ll see is “SeMachineAccountPrivilege” which will get you excited for a second until you realise what it does.

So I guess the point is, there’s some serious shell bell to be rung if you pay attention to this stuff.

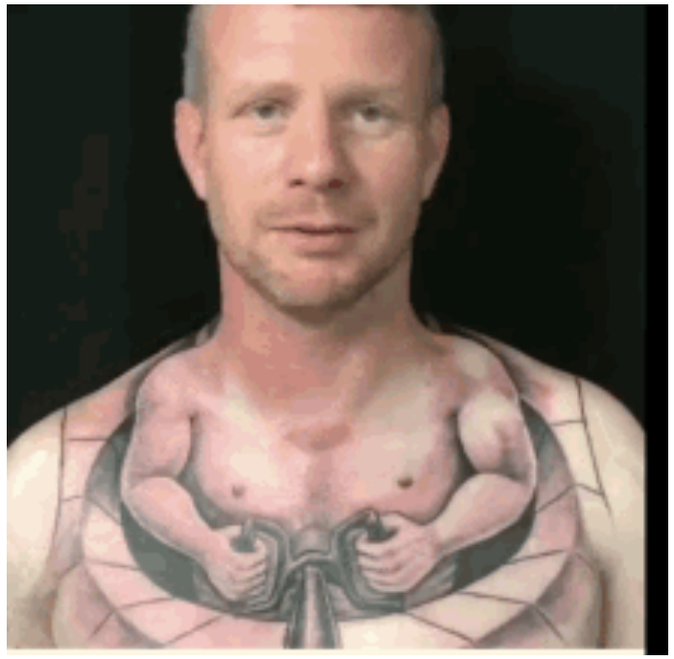
Regrets & Next Steps

- Should have used XPath queries.
- I want to further refine Grouper.
- You can probably help!

So the big lesson I learned is that I should have taken the time to read enough docco to learn how to use XPath queries in PowerShell. It would have saved me a lot of time.
A nice fella named ‘@liso’ I met on BloodHound slack is working on helping me rejig Grouper to use them.
Grouper still needs plenty more tweaking, in particular I could use a hand expanding which things
If you’re in a domain environment with policy templates that define abusable settings, or if you know of anything I’ve missed in the default stuff, I’d love to hear from you.
If you have ideas for removing the dependency on Microsoft’s Group Policy module and parsing the policy files straight from SYSVOL, I’d REALLY love to hear from you.

Thanks:

- the gang at Asterisk
- @sysop_host
- @prashant3535
- @harmj0y
- @liso
- the gang from the BloodHound Slack
- #ducksec



If I rushed this, there might be time for questions?

Bonus Content:

How do I know which policies apply to which users and computers?

TL;DR use PowerView

To see the policies:

```
Get-DomainGPO
```

To see the policies that apply to a given user/computer:

```
Get-DomainGPO -ComputerName testserver

Get-DomainGPO -UserName lowpriv
```

To see the computers that a policy applies to:

```
Get-DomainOU | WhereObject {$_.gplink.contains("{PUT THE GPO UID HERE}")} | %{Get-DomainComputer -ADSPath $_.distinguishedname}
```

To figure out which settings from which policies actually apply and which get overwritten…

OK so you've found a policy which is gonna make computers super vulnerable but you can't tell which machines it's going to apply to. How do you figure it out?

The simple version is: Get PowerView, get the GPO's UID using Get-DomainGPO, and do like this:

That'll tell you which OU(s) the policy is linked to. It's pretty rare in my experience to see people linking a GPO to multiple OUs but it can be done.

If you pipe the results from that into some more PowerView goodness, like this:

You should get a list of all computers to which a given policy applies. Same trick but with Get-DomainUser and you'll get the users it applies to. EZ.

Don’t bother trying to remember this:

Local GPOs are applied first.
Then GPOs linked to a site.
Then GPOs linked to the domain.
Then GPOs linked to OUs, from the top down.
If multiple GPOs are linked at the same level, they apply from the bottom-up.
Last writer of a given setting wins.

Well, it would be, but then there's the whole issue of policies overwriting one another, etc etc. I don't have an easy way to address that issue for you, but I can give you some useful info to help figure it out yourself:

WMI filtering and read permissions on the policy itself can also affect whether a policy applies to a given user/computer. WMI filtering usually boils down to targeting policy based on OS version, but it can target a bunch of other attributes of a host too.

Good luck I guess.