Active Directory Group Policy Troubleshooting

Lowell Vanderpool
TECH SAVVY PRODUCTIONS

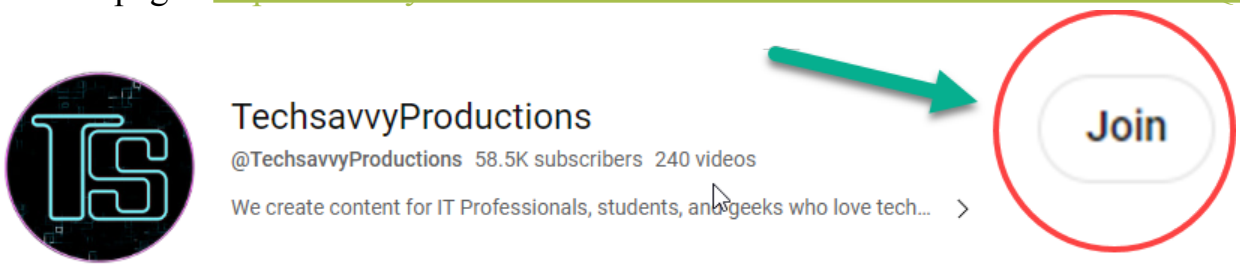
CONTACT US:

mrvanderpool@techsavvyproductions.com

SUPPORT US:

Please consider becoming a channel member:

- you get an early viewing of all our video content
- access to the complete series of videos for each subject
- links to video notes and PowerPoint slide deck both in MS-Word and PDF format
- Our eBook and resources folder
- Join our channel membership, it's $2.99/month); see the "Join" button on our channel homepage. https://www.youtube.com/channel/UCCAXBGYIJnScl0IFKXOIlsQ/join



"Everybody can be great... because anybody can serve. You don't have to have a college degree to serve. You don't have to make your subject and verb agree to serve. You only need a heart full of grace. A soul generated by love." Martin Luther King Jr.

.

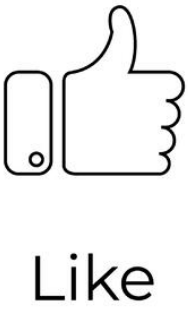
SOCIAL MEDIA AND WEBSITE:

Check out our YouTube channel for more content!
YouTube: https://www.youtube.com/user/vanderl2796/featured
Check out our Website: https://www.techsavvyproductions.com
Facebook: https://www.facebook.com/TechSavvyTeamFL
Twitter: https://twitter.com/vanderl2796
Telegram: https://t.me/Lowell901
Mr.V Linkedin: https://www.linkedin.com/in/lowell-vanderpool-57970623/
Email: mrvanderpool@techsavvyproductions.com

We translate subtitles on our videos into many languages:

Tech Savvy Productions

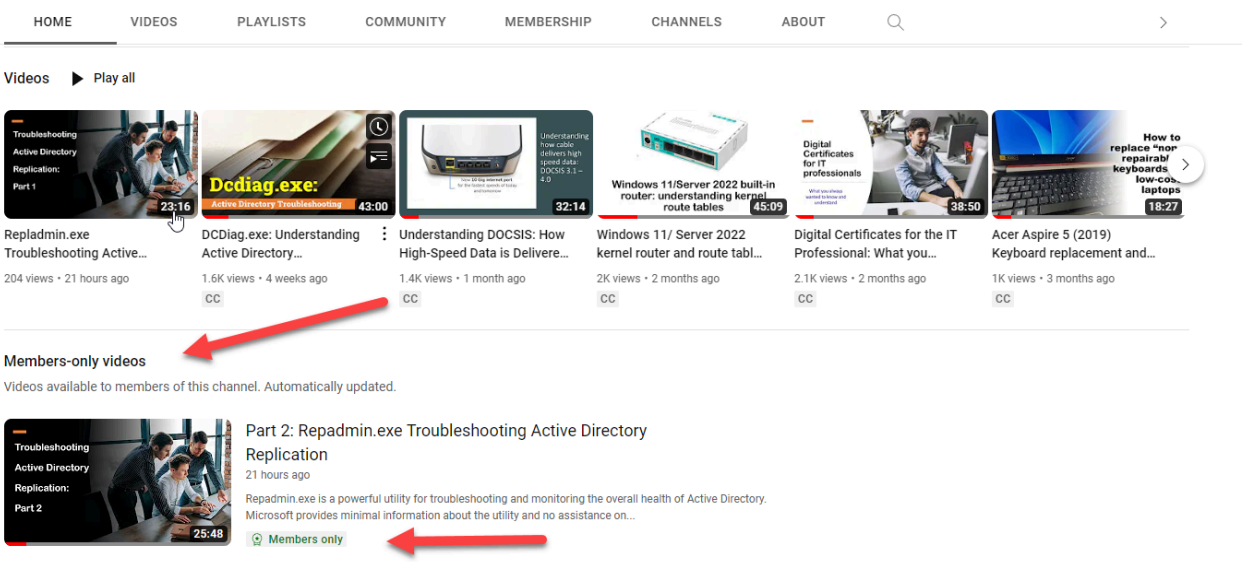
Two free ways to support our channel, like the video if it helped you better understand technology or the topic, and subscribe. Thank you for taking the time to do these helpful steps!



Where is member only resources?

Channel homepage and click on “Playlists” you will see a Members-only playlist.

Tech Savvy Productions



“I would maintain that thanks are the highest form of thought; and that gratitude is happiness doubled by wonder.”

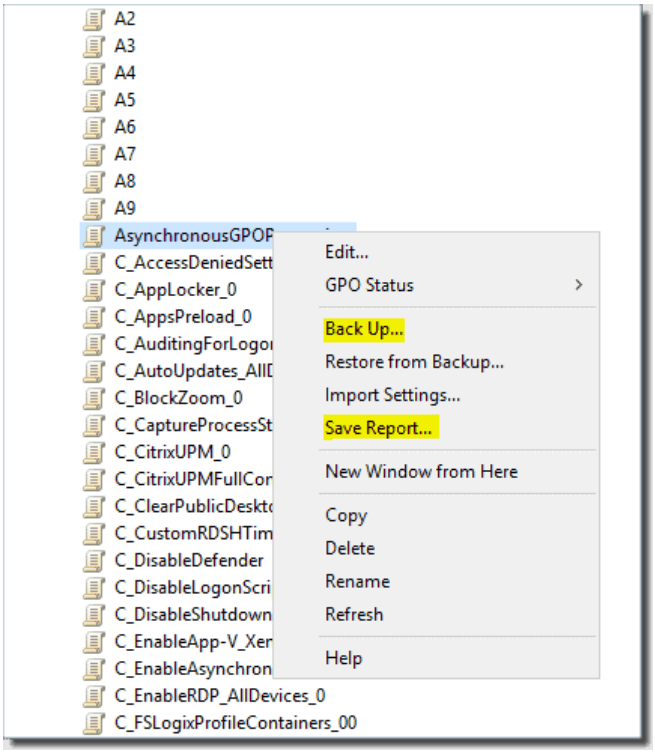
— G.K. Chesterton

Tech Savvy Productions

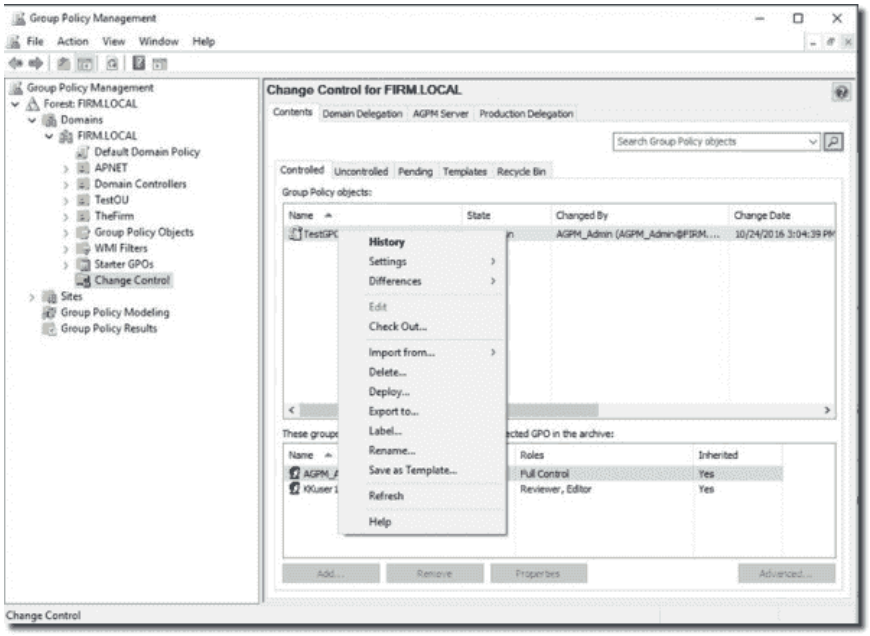
Table of Contents

Tech Savvy Productions

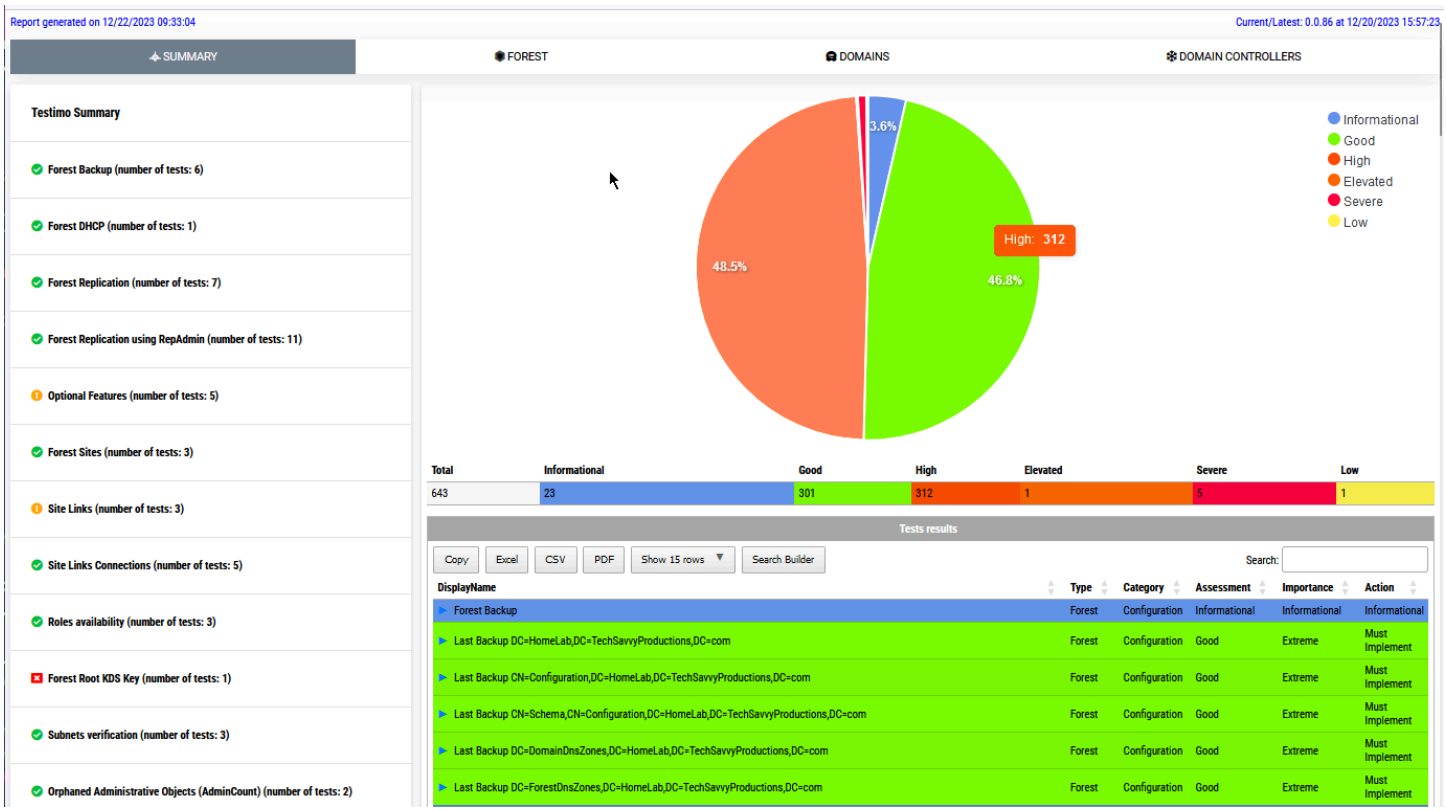
Tech Savvy Productions



Tech Savvy Productions



Tech Savvy Productions



Group Policy Objects (GPO) are primarily associated with Microsoft Windows domains and are used for managing the settings of computers and users within an Active Directory environment. While most tools for GPO are proprietary, there are open-source alternatives and complementary tools that can assist with GPO documentation and provide additional features. Here's a list of such open-source software:

1. **Ansible**: This is an open-source automation tool that can be used to automate the management of GPOs, although it requires some custom scripting and integration.
2. **Puppet**: Similar to Ansible, Puppet can be used for configuration management and can automate GPO tasks, though it also requires integration and custom scripting.
3. **Chef**: Another automation tool that can be used for managing infrastructure, including GPOs, with custom scripting.
4. **Terraform**: Known for infrastructure as code, Terraform can be used to manage cloud infrastructure, which can indirectly affect GPO management.
5. **SaltStack**: This tool is used for automation, configuration management, and orchestration, and can be adapted for GPO management.
6. **Git**: While not a direct GPO tool, Git can be used for version control of GPO scripts and policies.
7. **Zentyal**: An open-source server that can act as a domain controller and manage GPOs for small and medium networks.
8. **WPKG**: It's an automated software deployment, upgrade, and removal program for Windows. It can be used to ensure that GPO-relevant software is installed across a network.
9. **PolicyD**: Often used in mail server configurations, it can be adapted for broader policy enforcement.
10. **GPOZaurr**: A PowerShell module that helps in documenting, reviewing, and managing GPOs, although it requires PowerShell knowledge.
11. **PowerShell DSC (Desired State Configuration)**: While a part of PowerShell, DSC allows for configuration management that can align with GPO settings.
12. **Sysvol Explorer**: Useful for exploring and understanding the contents of the SYSVOL folder in a domain, which is where GPOs are stored.

Tech Savvy Productions

Group Policy Eater

is a PowerShell module that aims to gather information about Group Policies but also allows fixing issues that you may find in them. **The only command you will ever need to understand and fix your Group Policies (GPO)**

https://github.com/EvotecIT/GPOZaurr

I've been working on cleaning up **Group Policies** for a couple of months. While it may seem trivial, things get complicated when you're tasked with managing 5000 **GPOs** created over 15 years by multiple teams without any best practices in mind. While working on **GPOZaurr** (my new **PowerShell** module), I've noticed that the more code I wrote to manage those **GPOs**, the more I knew passing this knowledge to admins who will be executing this on a **weekly/monthly basis** is going to be a challenge. That's why I've decided to follow a similar approach as my other Active Directory testing module called **Testimo**. I've created a single command that analyses **Group Policies** using different methods and shows views from different angles to deliver the full picture. On top of that, it provides a solution (or it tries to) so that it's fairly easy to fix – as long as you agree with what it proposes.

Please be careful when using this on production

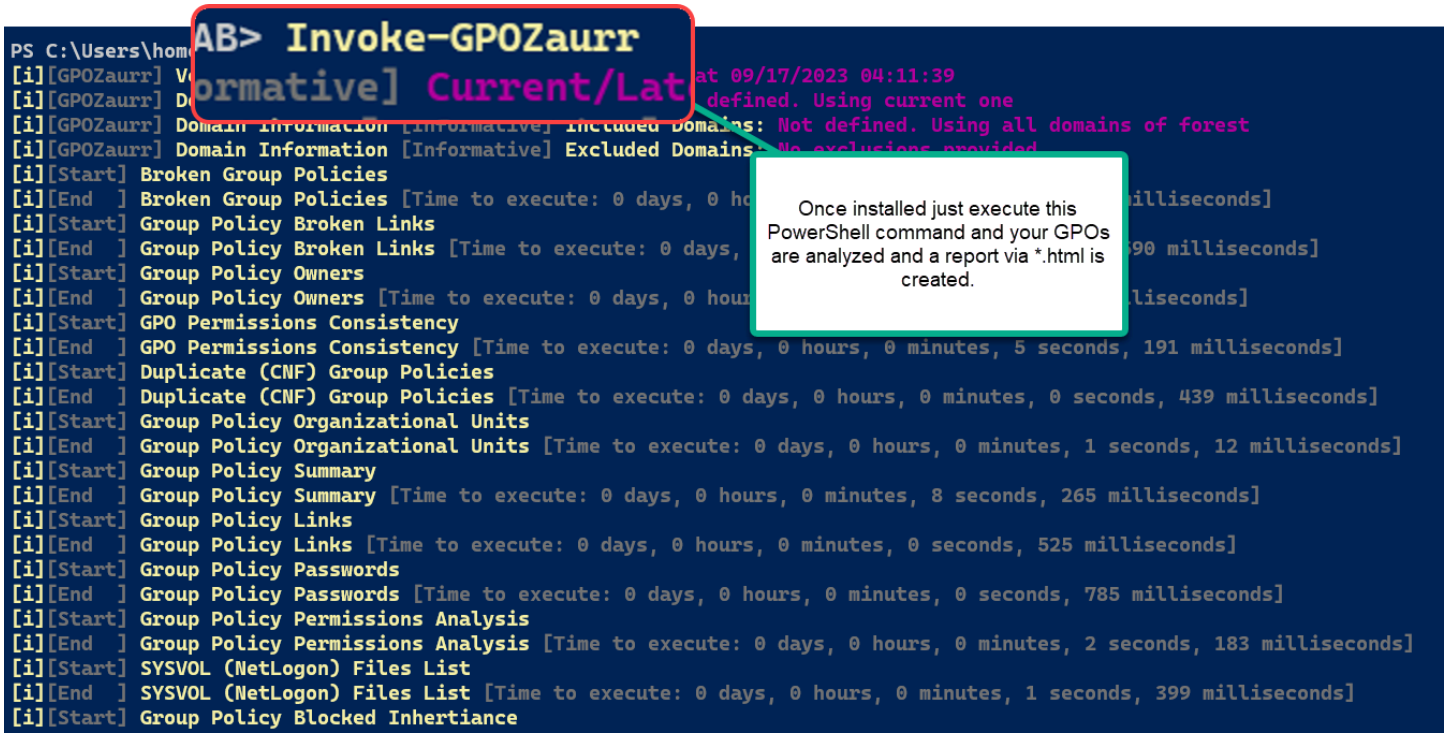
I've done a lot of research and put a lot of effort into making sure this **PowerShell** module works as expected. However, I do make mistakes. Contrary to my usual work, this module is not read-only. To almost every read command, there is also a set or remove command. It can change things, delete them, or modify them. If you don't understand what will happen, don't do it. Review source code, run read commands first to understand the output, what it's showing. If you have doubts – don't use it or create an issue on **GitHub** to clarify. All cmdlets that have the ability to write/delete contain **WhatIf/LimitProcessing** count parameters. Use them before implementing any changes!

Please keep in mind I've tested GPOZaurr only on English based Active Directory. I have no clue how it will behave on non-English systems. As I've not worked with other languages for a while, I don't remember if object types are still reported in English by PowerShell or reported in language equivalent. Be careful.

Tech Savvy Productions

GPOZaurr

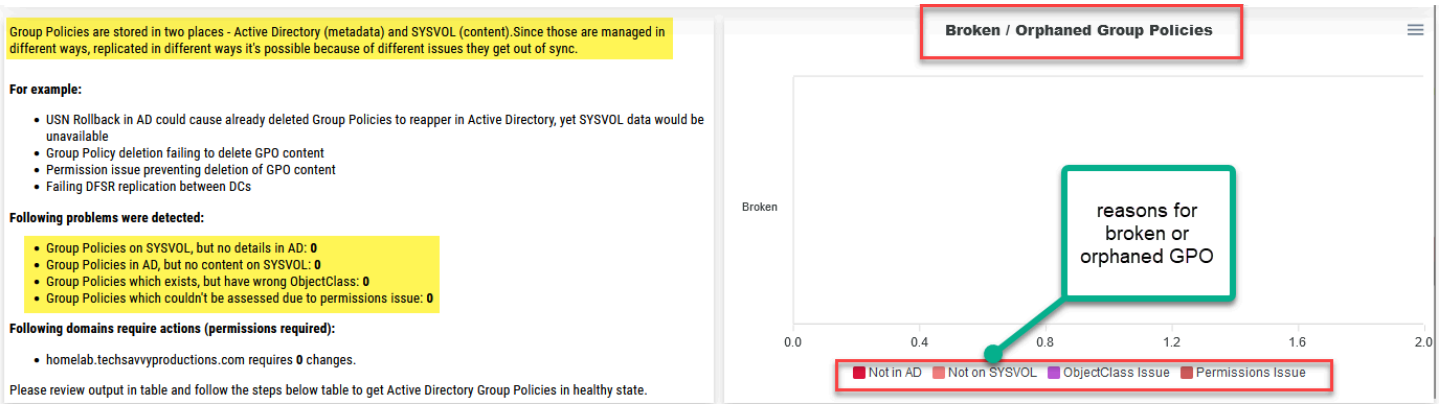
Group Policy Eater is a PowerShell module that aims to gather information about Group Policies but also allows fixing issues that you may find in them. **GPOZaurr** provides 360 degrees of information about Group Policies and their settings.



Just a single command (`aa`) provides following reports:

- GPOBroken
- GPOBrokenLink

When GPO is deleted correctly, it usually is removed from AD, SYSVOL, and any link to it is also discarded.
Unfortunately, this is true only if the GPO is created and linked within the same domain. If GPO is linked in another domain, this leaves a broken link hanging on before it was linked.
Additionally, the Remove-GPO cmdlet doesn't handle site link deletions, which causes dead links to be stuck on sites until those are manually deleted. This means that any GPOs deleted using PowerShell may leave a trail.
As it stands currently there are **0** broken links that need to be deleted over **0** unique objects.



- GPOOwners

By default, GPO creation is usually maintained by Domain Admins or Enterprise Admins.

When GPO is created by Domain Admins or Enterprise Admins group members, the GPO Owner is set to Domain Admins.

When GPO is created by a member of Group Policy Creator Owners or other group has delegated rights to create a GPO, the owner of said GPO is not Domain Admins group but is assigned to the relevant user.

GPO Owners should be Domain Admins or Enterprise Admins to prevent abuse.

If that isn't so, it means the owner can fully control GPO and potentially change its settings in an uncontrolled way.

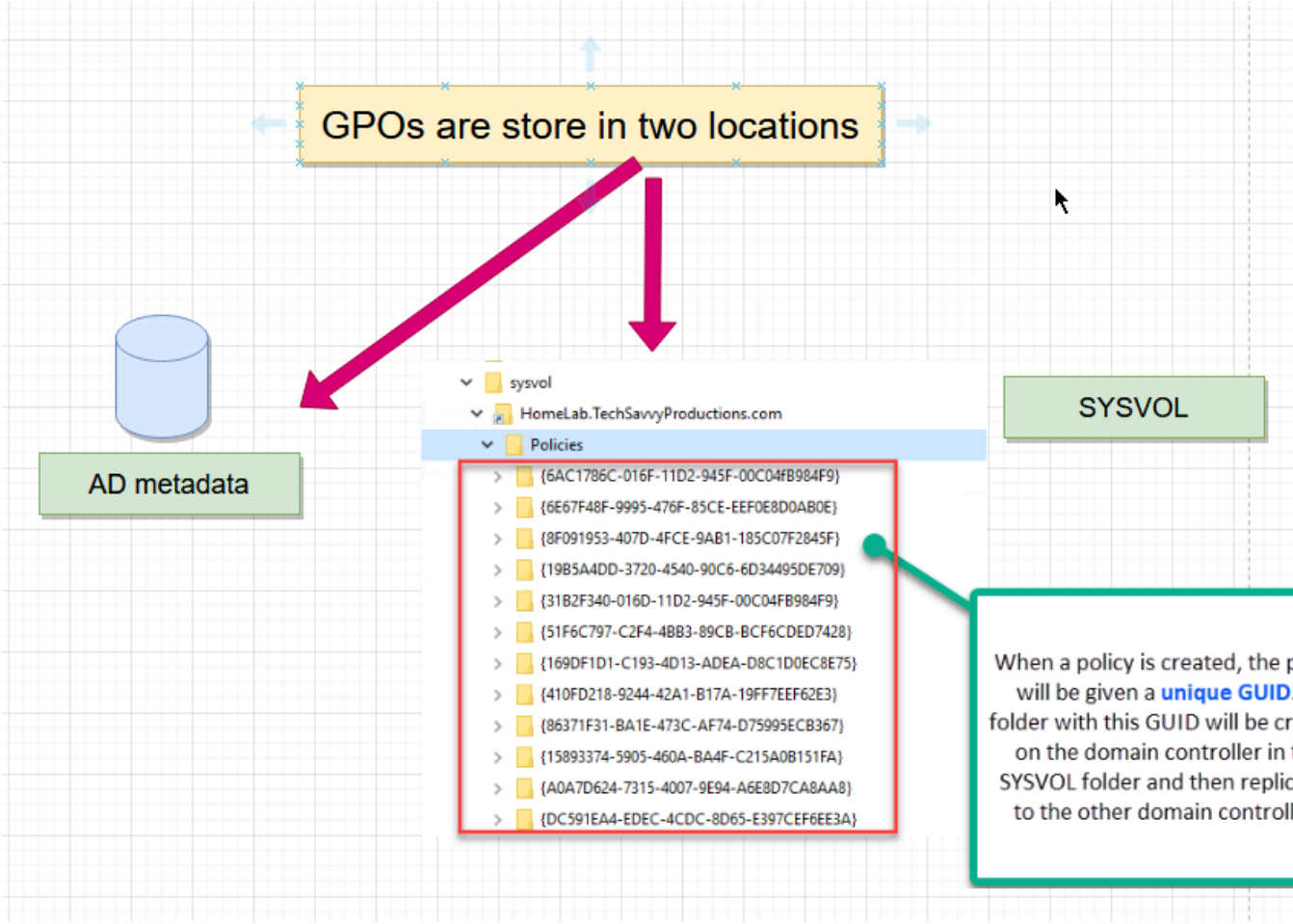
While at the moment of creation of new GPO, it's not a problem, in the long term, it's possible such a person may no longer be admin, yet keep their rights over GPO.

As your aware, Group Policies are stored in 2 places. In Active Directory (metadata) and SYSVOL (settings).

This means that there are 2 places where GPO Owners exists. This also means that for multiple reasons, AD and SYSVOL can be out of sync when it comes to their permissions, which can lead to uncontrolled ability to modify them.

Ownership in Active Directory and Ownership of SYSVOL for said GPO is required to be the same.

Tech Savvy Productions



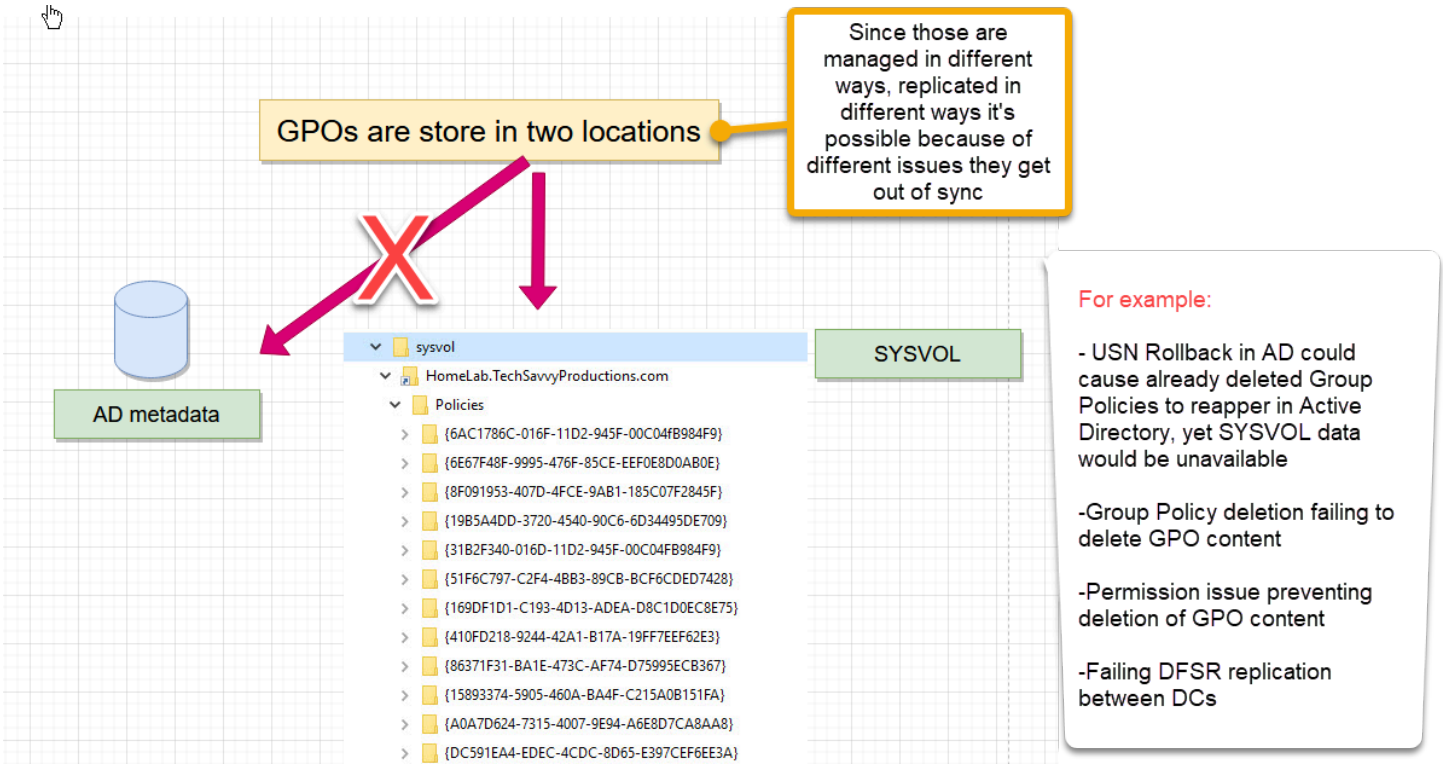
Here's a short summary of **Group Policy Owners**:

- Administrative Owners: **13**
- Non-Administrative, but approved Owners (for example AGPM): **0**
- Non-Administrative Owners: **2**
- Owners consistent in AD and SYSVOL: **13**
- Owners not-consistent in AD and SYSVOL: **2**

Following will need to happen:

- Group Policies requiring owner change: **2**
- Group Policies which can't be fixed (no SYSVOL?): **0**
- Group Policies unaffected: **13**

Tech Savvy Productions

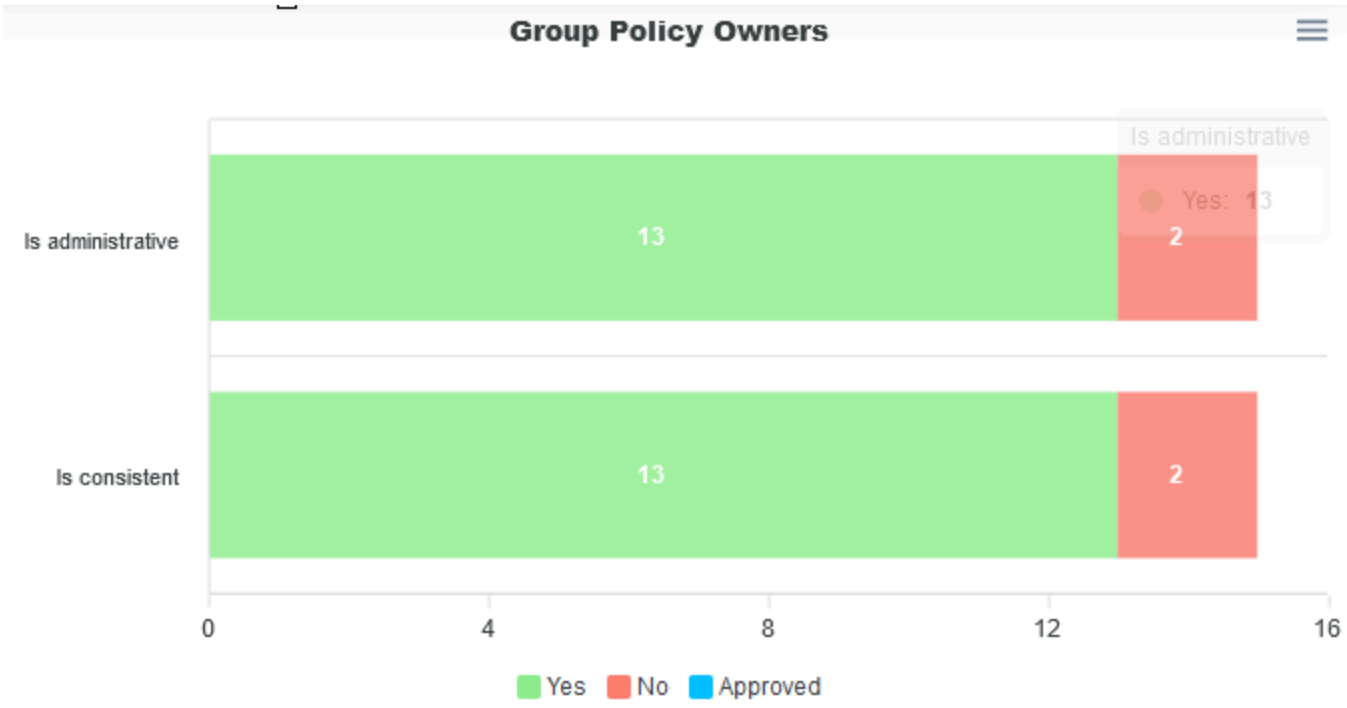


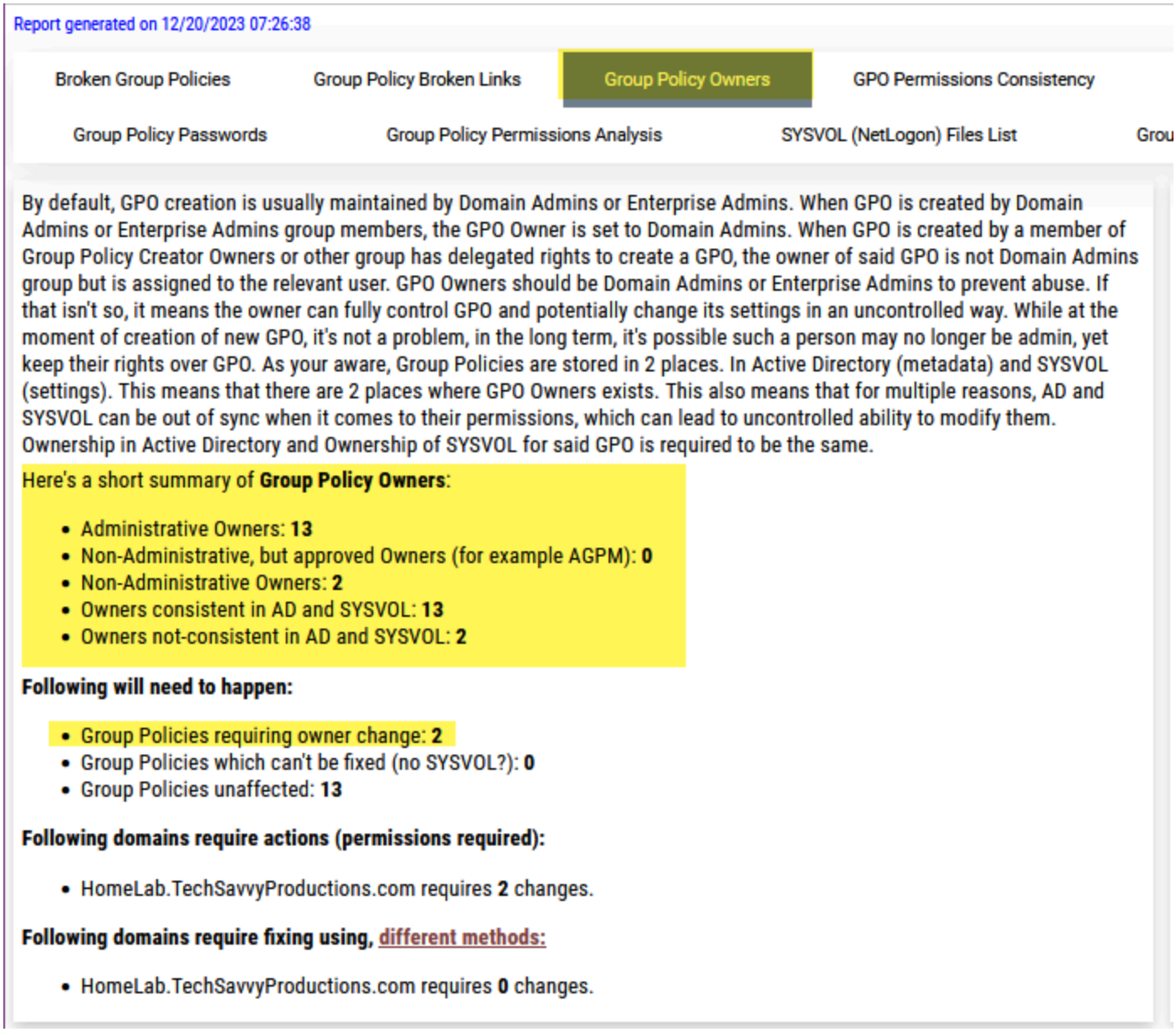
Following domains require actions (permissions required):

- HomeLab.TechSavvyProductions.com requires **2** changes.

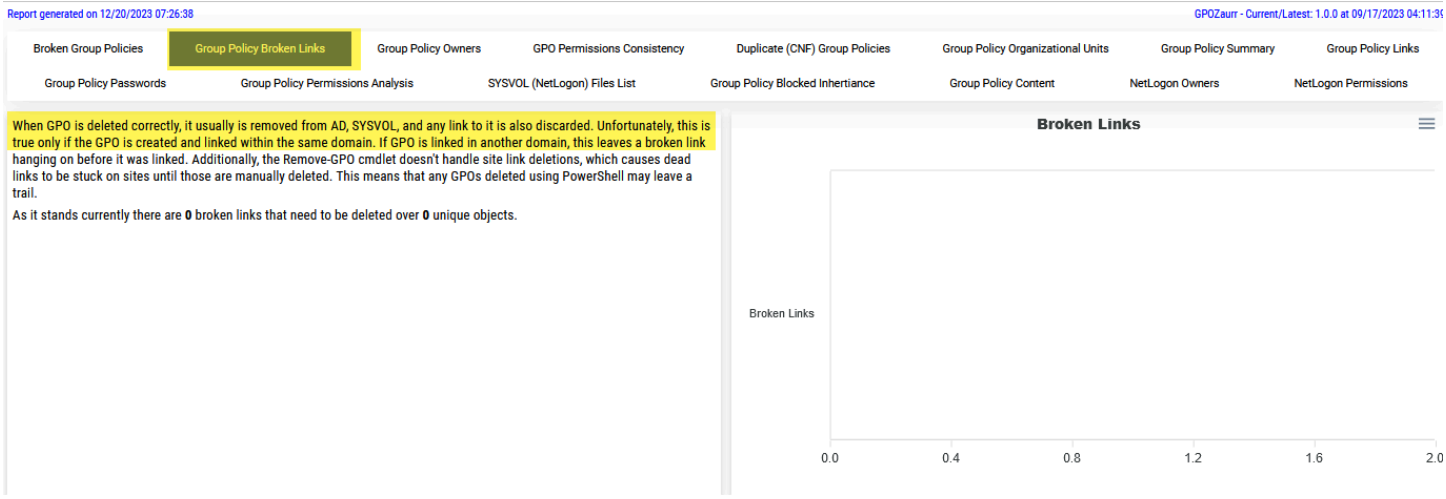
Following domains require fixing using, <u>different methods:</u>

- HomeLab.TechSavvyProductions.com requires **0** changes.


Group Policy Owners
Is administrative
Yes: 13
Is administrative
13
2
Is consistent
13
2
0
4
8
12
16
Yes
No
Approved



- GPOConsistency
- GPODuplicates
- GPOOrganizationalUnit
- GPOList
- GPOLinks

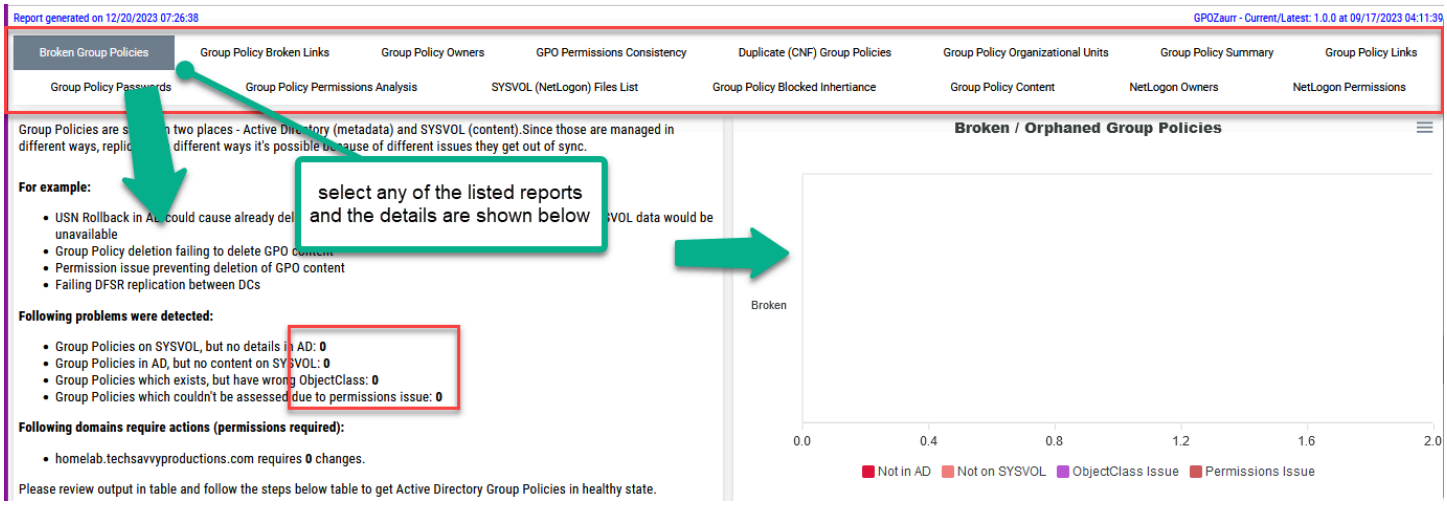


- GPOPassword
- GPOPermissions
- GPOPermissionsAdministrative
- GPOPermissionsRead
- GPOPermissionsRoot
- GPOPermissionsUnknown
- GPOFiles
- GPOBlockedInheritance
- GPOAnalysis
- GPOUpdates
- NetLogonOwners
- NetLogonPermissions
- SysVolLegacyFiles

But that's not all. There are over 50 other commands available that make it even more powerful helping with day to day tasks to manage Group Policies.

To understand the usage of `Invoke-GPOZaurr` I've created blog post you may find useful

Tech Savvy Productions



- The only command you will ever need to understand and fix your Group Policies (GPO)

Installing

GPOZaurr requires RSAT installed to provide results. If you don't have them you can install them as below. Keep in mind it also installs GUI tools so it shouldn't be installed on user workstations.

Windows 10 Latest

```
Add-WindowsCapability -Online -Name
'Rsat.ActiveDirectory.DS-LDS.Tools~~~~0.0.1.0'
```

```
Add-WindowsCapability -Online -Name
'Rsat.GroupPolicy.Management.Tools~~~~0.0.1.0'
```

Finally just install module:

```
Install-Module -Name GPOZaurr -AllowClobber -Force
```

Force and AllowClobber aren't necessary, but they do skip errors in case some appear.

Updating

```
Update-Module -Name GPOZaurr
```

That's it. Whenever there's a new version, you run the command, and you can enjoy it. Remember that you may need to close, reopen PowerShell session if you have already used module before updating it.

The essential thing is if something works for you on production, keep using it till you test the new version on a test computer. I do changes that may not be big, but big enough that auto-update may break your code. For example, small rename to a parameter and your code stops working! Be responsible!

DisplayName	Status	DomainName	SysvolServer	ObjectClass	Id	Path
BitLocker	Exists	homelab.techsavvyproductions.com	homelab.techsavvyproductions.com	groupPolicyContainer	{169DF1D1-C193-4D13-ADEA-D8C1D0EC8E75}	\\homelab.techsavvyproductions.com\SYSVOL\homelab.techsavvyproductions.com\Policies\{169DF1D1-C193-4D13-ADEA-D8C1D0EC8E75}
Certificates	Exists	homelab.techsavvyproductions.com	homelab.techsavvyproductions.com	groupPolicyContainer	{410FD218-9244-42A1-B17A-19FF7EEF62E3}	\\homelab.techsavvyproductions.com\SYSVOL\homelab.techsavvyproductions.com\Policies\{410FD218-9244-42A1-B17A-19FF7EEF62E3}
Default Domain Controllers Policy	Exists	homelab.techsavvyproductions.com	homelab.techsavvyproductions.com	groupPolicyContainer	{6AC1786C-016F-11D2-945F-00C04fB984F9}	\\homelab.techsavvyproductions.com\SYSVOL\homelab.techsavvyproductions.com\Policies\{6AC1786C-016F-11D2-945F-00C04fB984F9}
Default Domain Policy	Exists	homelab.techsavvyproductions.com	homelab.techsavvyproductions.com	groupPolicyContainer	{31B2F340-016D-11D2-945F-00C04FB984F9}	\\homelab.techsavvyproductions.com\SYSVOL\homelab.techsavvyproductions.com\Policies\{31B2F340-016D-11D2-945F-00C04FB984F9}
Disable Anonymous SID Enumeration	Exists	homelab.techsavvyproductions.com	homelab.techsavvyproductions.com	groupPolicyContainer	{F3B576C2-63CC-4AAE-9175-D08BCE1CBF43}	\\homelab.techsavvyproductions.com\SYSVOL\homelab.techsavvyproductions.com\Policies\{F3B576C2-63CC-4AAE-9175-D08BCE1CBF43}
Disable Guest Account	Exists	homelab.techsavvyproductions.com	homelab.techsavvyproductions.com	groupPolicyContainer	{19B5A4DD-3720-4540-90C6-6D34495DE709}	\\homelab.techsavvyproductions.com\SYSVOL\homelab.techsavvyproductions.com\Policies\{19B5A4DD-3720-4540-90C6-6D34495DE709}

“It is absurd for the Evolutionist to complain that it is unthinkable for an admittedly unthinkable God to make everything out of nothing, and then pretend that it is more thinkable that nothing should turn itself into everything.”

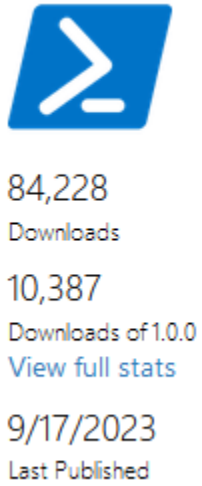
— G.K. Chesterton

The only command you will ever need to understand and fix your Group Policies (GPO)

GPOZaurr

1.0.0

Group Policy Eater is a PowerShell module that aims to gather information about Group Policies but also allows fixing issues that you may find in them.



I've been working on cleaning up **Group Policies** for a couple of months. While it may seem trivial, things get complicated when you're tasked with managing 5000 **GPOs** created over 15 years by multiple teams without any best practices in mind. While working on GPOZaurr (my new **PowerShell** module), I've noticed that the more code I wrote to manage those **GPOs**, the more I knew passing this knowledge to admins who will be executing this on a **weekly/monthly basis** is going to be a challenge. That's why I've decided to follow a similar approach as my other Active Directory testing module called Testimo. I've created a single command that analyses **Group Policies** using different methods and shows views from different angles to deliver the full picture. On top of that, it provides a solution (or it tries to) so that it's fairly easy to fix – as long as you agree with what it proposes.

Please be careful when using this on production

I've done a lot of research and put a lot of effort into making sure this **PowerShell** module works as expected. However, I do make mistakes. Contrary to my usual work, this module is not read-only. To almost every read command, there is also a set or remove command. It can change things, delete them, or modify them. If you don't understand what will happen, don't do it. Review source code, run read commands first to understand the output, what it's showing. If you have doubts – don't use it or create an issue on **GitHub** to clarify. All cmdlets that have the ability to write/delete contain **WhatIf/LimitProcessing** count parameters. Use them before implementing any changes!

Please keep in mind I've tested GPOZaurr only on English based Active Directory. I have no clue how it will behave on non-English systems. As I've not worked with other languages for

Tech Savvy Productions

a while, I don't remember if object types are still reported in English by PowerShell or reported in language equivalent. Be careful.

Useful Links

Please make sure to visit **GitHub** to review sources or report issues. If you're going to use it, I recommend doing it via **PowerShellGallery** as that version is minimized and optimized. Reviewing sources is easier on the GitHub version as it has more comments and is divided into sections.

The code is published on GitHub

Issues should be reported on GitHub

Code is published as a module on PowerShellGallery

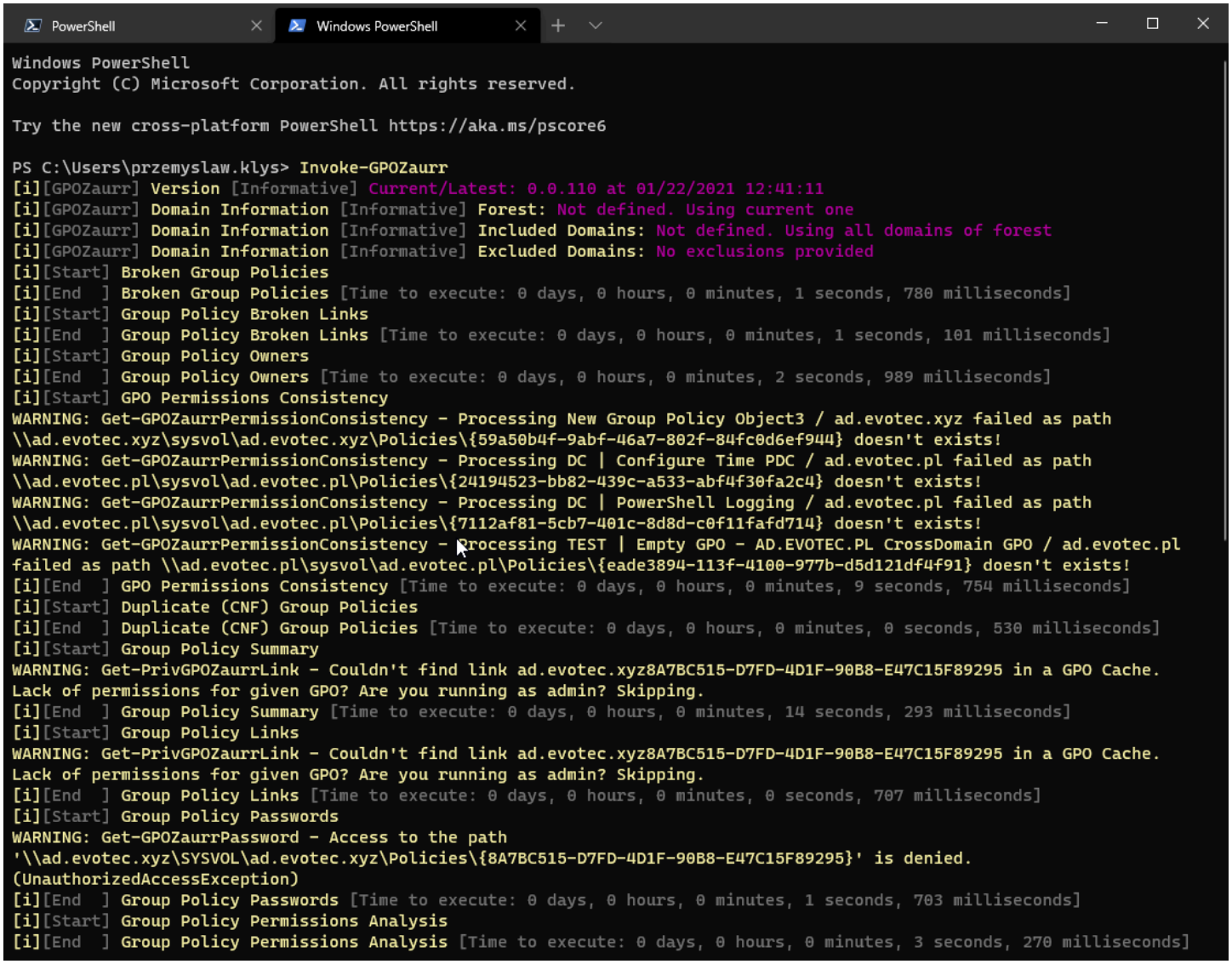
The module is signed with a certificate, like any new modules that I create or update.

1. Install-Module GPOZaurr -Force

Invoke-GPOZaurr - One command that makes a difference

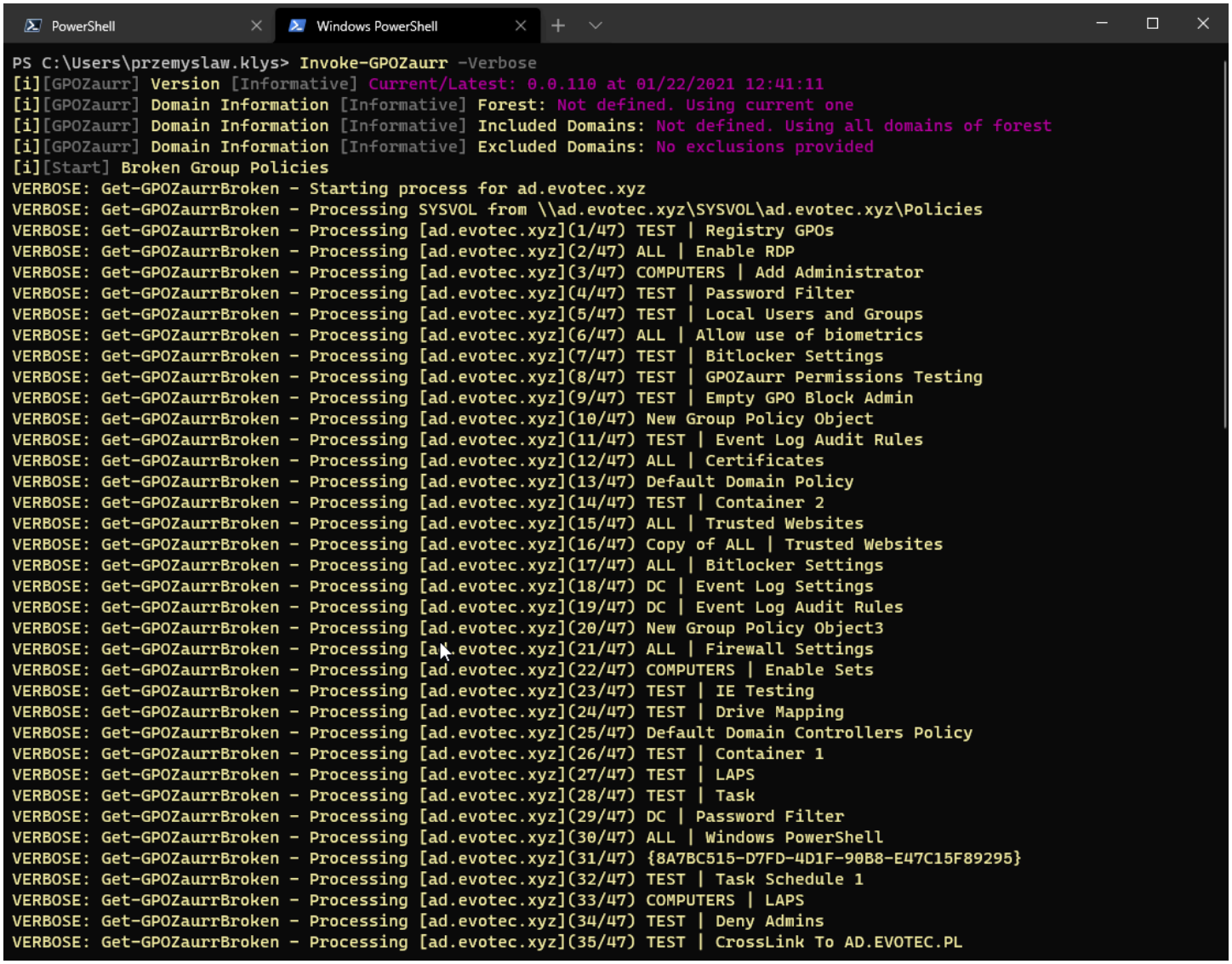
As mentioned before, **Invoke-GPOZaurr** follows a similar pattern to what **Invoke-Testimo** does. When run without any parameters, it will go thru all available reports one by one to deliver a full-scope scan. Keep in mind that running this cmdlet without any parameters is fine for small domains, but it will take hours to complete for larger domains. For the domain of 5000 GPOs, some reports can take even 2 hours to complete.

Tech Savvy Productions



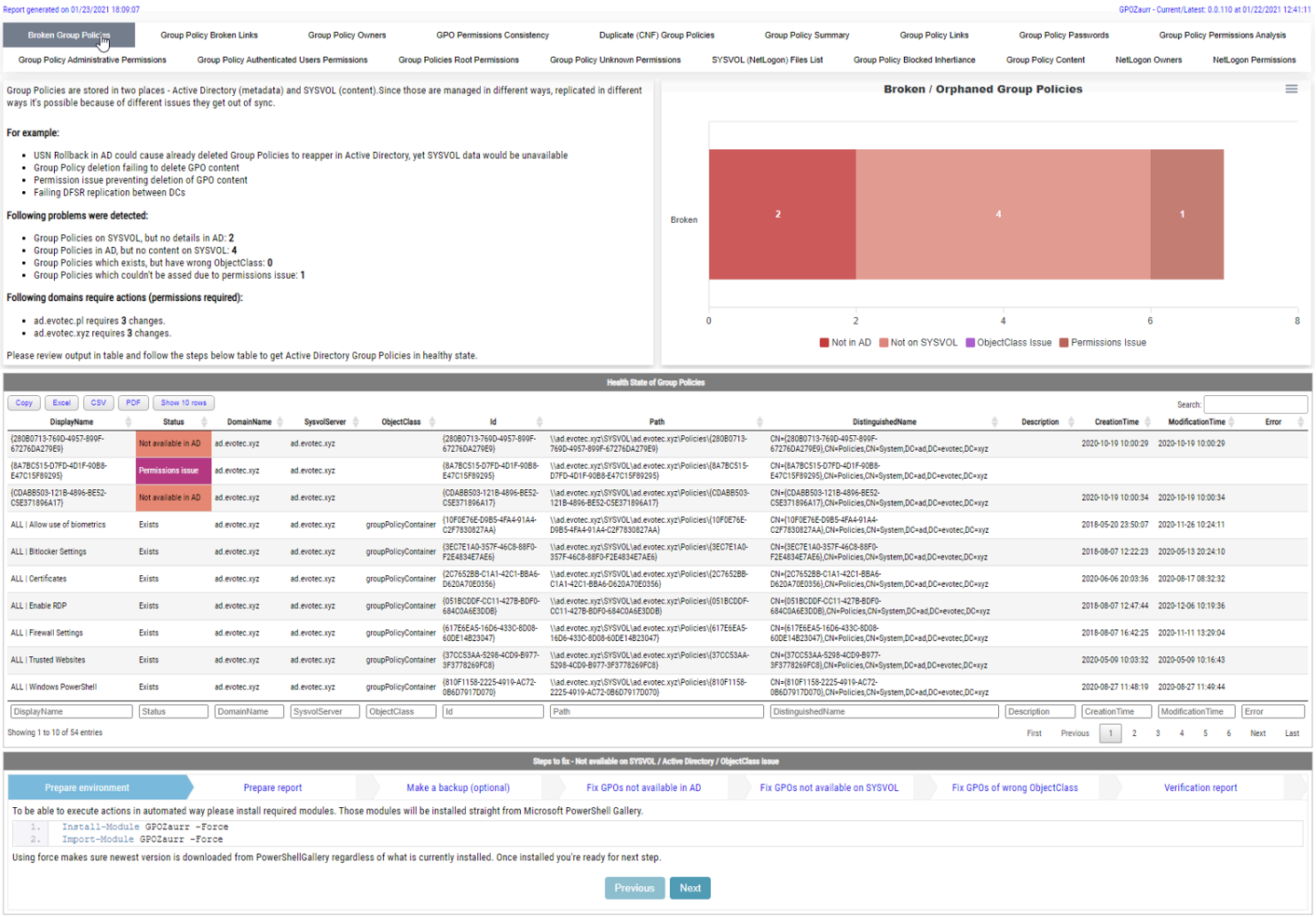
When run, it will display a short information about what it is currently doing and which report is being generated. If you have a large domain and things take time, you may want to use **Invoke-GPOZaurr** with **Verbose** parameter to get additional information.

Tech Savvy Productions



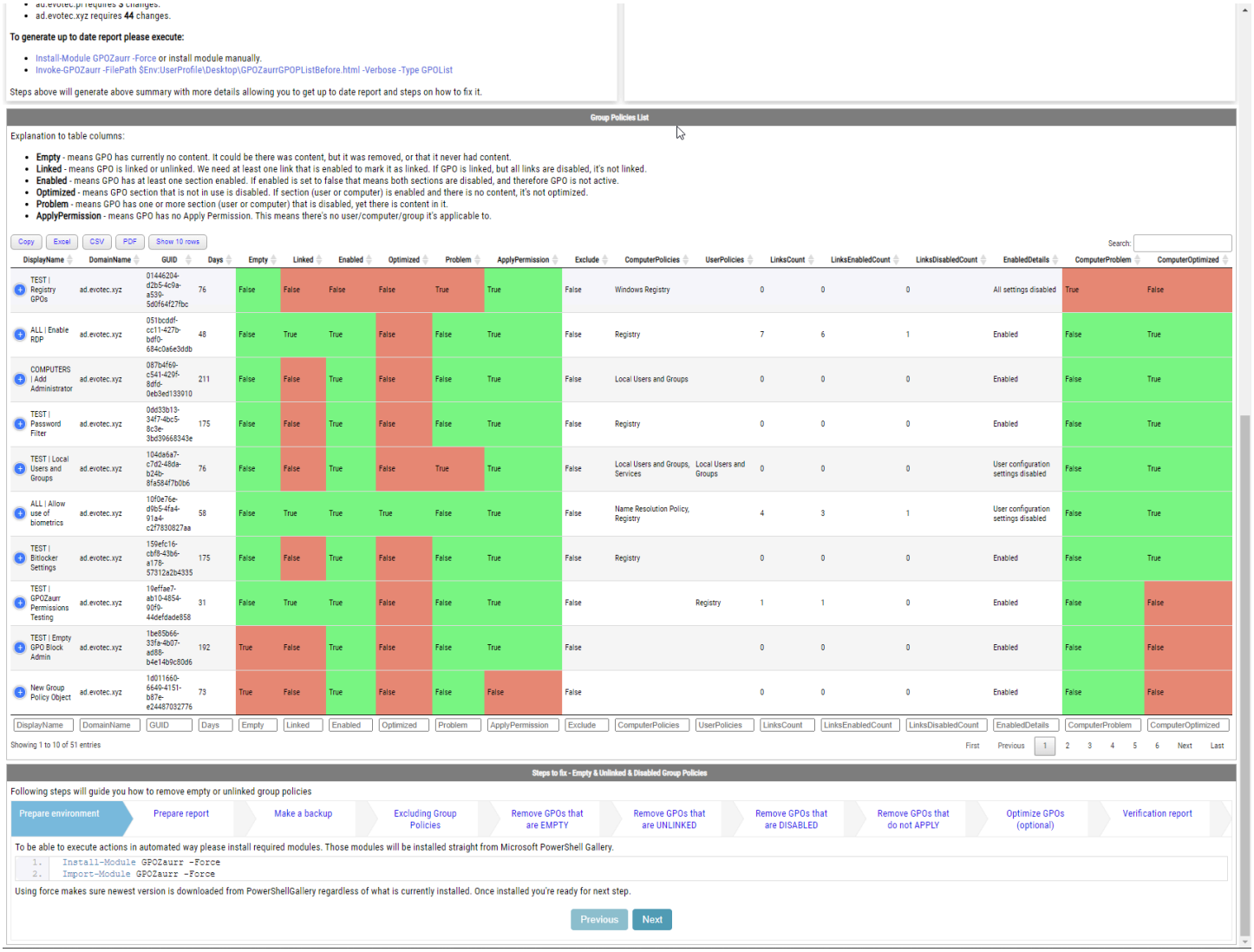
Once the cmdlet is complete HTML report will open up automatically.

Tech Savvy Productions



As you can see on the screenshot above, multiple reports were created, each on a different tab. The design of the report is mostly the same. There is information about what the report detected and why it did so on the report's top left. It also gives you a summary of your whole forest and where the issues are found. In the top right corner, I've added a small chart that visualizes the current status. Some charts will show only problems. Some will show multiple statues – all depending on the type of report getting generated. There is usually one, but sometimes more tables with displayed information depending on the problem in the second section. Tables are color-coded to visualize better what is bad or to distinguish multiple problems within the same report. Tables also allow you to export data to Excel, CSV or PDF.

Tech Savvy Productions



Finally, the last section contains the solution to the problem described. It usually provides step by step instructions on fixing the problem if you choose to fix it. Most of the time, solutions are automated to the point where a single line of code can fix an issue. For example, **delete all empty GPOs**, **delete all unlinked GPOs**, and so on. One command, zero effort.

Invoke-GPOZaurr - Available reports

Currently, **Invoke-GPOZaurr** has few built-in reports. Some of them are more advanced, some of them are for review only. Here's the full list for today. Not everything is 100% finished. Some will require some updates soon as I get more time and feedback. Feel free to report issues/improve those reports with more information.

- **GPOBroken** – this report can detect GPOs that are broken. By broken GPOs, I mean those which exist in AD but have no SYSVOL content or vice versa – have SYSVOL content, but there's no AD metadata. Additionally, it's able to detect GPC objects that are

Tech Savvy Productions

no longer GroupPolicy object. – Then, it provides an easy way to fix it using given step-by-step instructions.

- **GPOBrokenLink** – this report can detect links that have no matching GPO. For example, if a GPO is deleted, sometimes links to that GPO are not properly removed. This command can detect that and propose a solution.
- **GPOOwners** – this report focuses on GPO Owners. By design, if Domain Admin creates GPO, the owner of GPO is the domain admins group. This report detects GPOs that are not owned by Domain Admins (in both SYSVOL and AD) and provides a way to fix them.
- **GPOConsistency** – this report detects inconsistent permissions between Active Directory and SYSVOL, verifying that files/folders inside each GPO match permissions as required. It then provides you an option to fix it.
- **GPODuplicates** – this report detects GPOs that are CNF, otherwise known as duplicate AD Objects, and provides a way to remove them.
- **GPOList** – this report summarizes all group policies focusing on detecting Empty, Unlinked, Disabled, No Apply Permissions GPOs. It also can detect GPOs that are not optimized or have potential problems (disabled section, but still settings in it)
- **GPOLinks** – this report summarizes links showing where the GPO is linked, whether it's linked to any site, cross-domain, or the status of links.
- **GPOPassword** – this report should detect passwords stored in GPOs.
- **GPOPermissions** – this report provides full permissions overview for all GPOs. It detects GPOs missing read permissions for Authenticated Users, GPOs that miss Domain Admins, Enterprise Admins, or SYSTEM permissions. It also detects GPOs that have Unknown permissions available. Finally, it allows you to fix permissions for all those GPOs easily. It's basically a one-stop for all permission needs.
- **GPOPermissionsAdministrative** – this report focuses only on detecting missing Domain Admins, Enterprise Admins permissions and allows you to fix those in no time.
- **GPOPermissionsRead** – similar to an administrative report, but this one focuses on Authenticated Users missing their permissions.
- **GPOPermissionsRoot** – this report shows all permissions assigned to the root of the group policy container. It allows you to verify who can manage all GPOs quickly.

Tech Savvy Productions

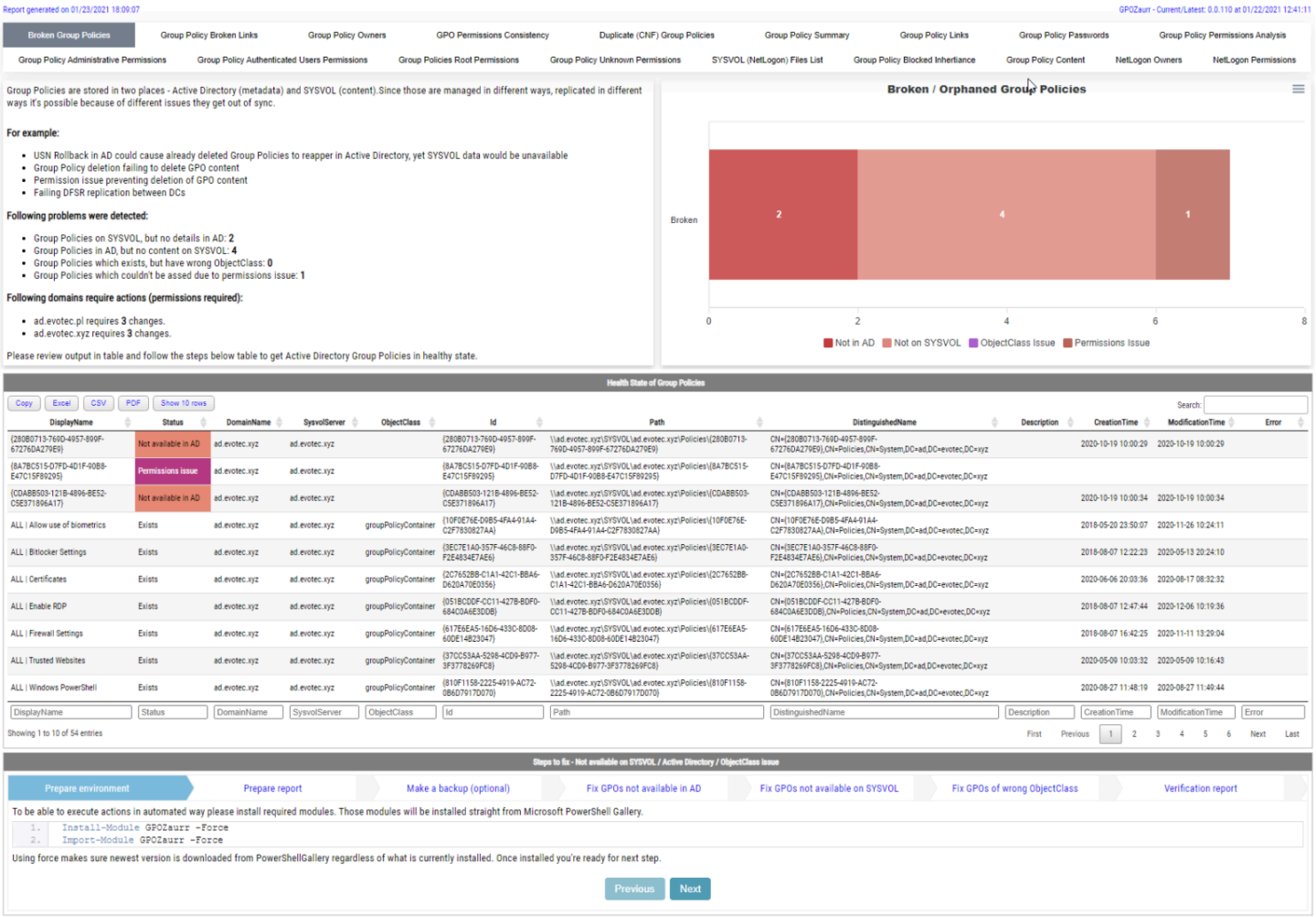
- **GPOPermissionsUnknown** – this report focuses on detecting unknown permissions (deleted users) and allows you to remove them painlessly.
- **GPOFiles** – this report lists all files in the SYSVOL folder (including hidden ones) and tries to make a decent guess whether the file placement based on extension/type makes sense or requires additional verification. This was written to find potential malware or legacy files that can be safely deleted.
- **GPOBlockedInheritance** – this report checks for all Organizational Units with blocked inheritance and verifies the number of users or computers affected.
- **GPOAnalysis** – this report reads all content of group policies and puts them into 70+ categories. It can show things like GPOs that do Drive Mapping, Bitlocker, Laps, Printers, etc. It's handy to find dead settings, dead hosts, or settings that no longer make sense.
- **NetLogonOwners** – this report focuses on detecting NetLogon Owners and a way to fix it to default, secure values.
- **NetLogonPermissions** – this report provides an overview and assessment of all permissions on the NetLogon share.
- **SysVolLegacyFiles** – this report detects SYSVOL Legacy Files (.adm) files

```
Invoke-GPOZaurr - Report GPOBroken
```

Group Policies are stored in two places – Active Directory (metadata) and SYSVOL (content). Since those are managed in different ways, replicated in different ways, it's possible because of different issues, and they get out of sync.

1. Invoke-GPOZaurr -Type GPOBroken

Tech Savvy Productions

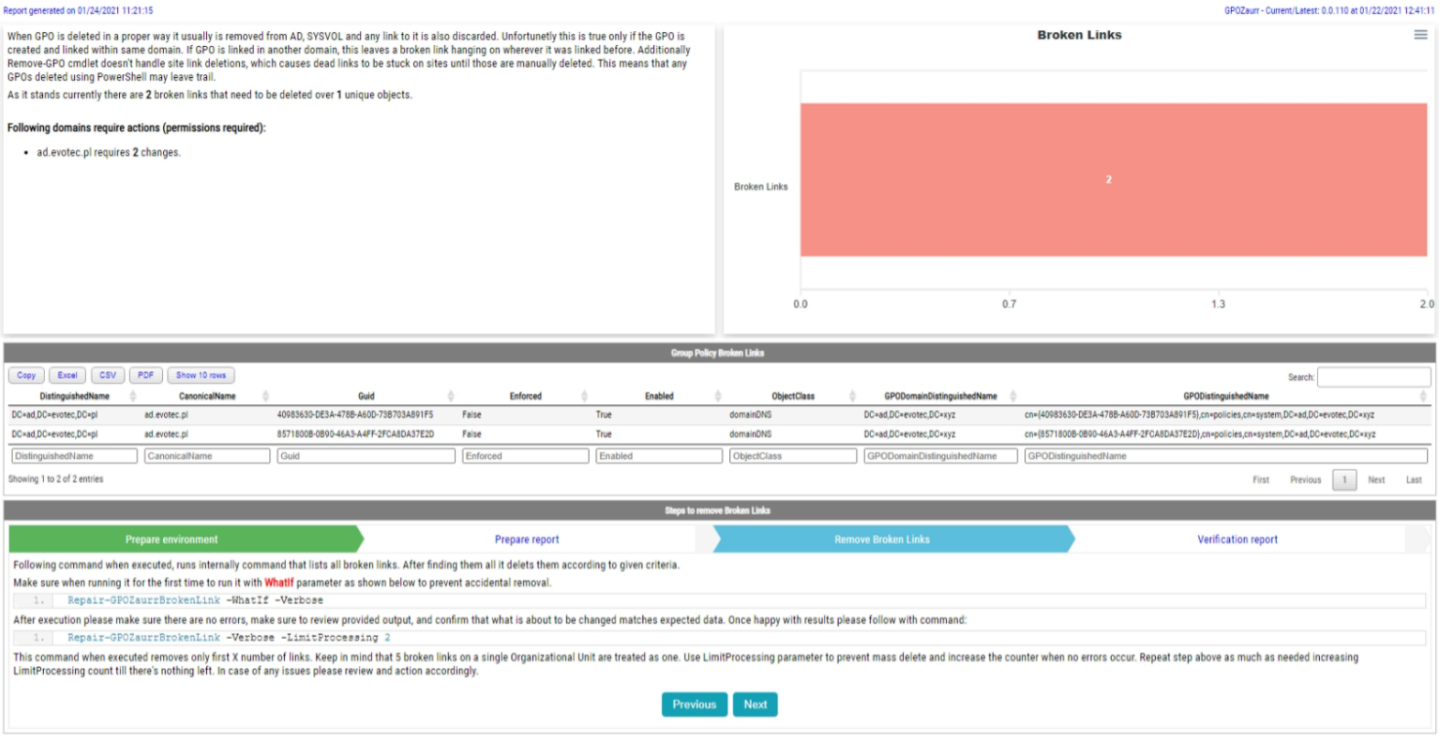


With just a few simple steps, you can have that fixed in a couple of minutes. Keep in mind that you need to have healthy replication of group policies for this to work and not report false positives. If you have unhealthy replication and wrong, DC will get asked about those issues you could potentially remove legitimate content.

Invoke-GPOZaurr - Report GPOBrokenLink

When GPO is deleted correctly, it usually is removed from AD, SYSVOL, and any link to it is also discarded. Unfortunately, this is true only if the GPO is created and linked within the same domain. If GPO is linked in another domain, this leaves a broken link hanging on before it was linked. Additionally, the Remove-GPO cmdlet doesn't handle site link deletions, which causes dead links to be stuck on sites until those are manually deleted. This means that any GPOs deleted using PowerShell may leave a trail.

1. Invoke-GPOZaurr -Type GPOBrokenLink

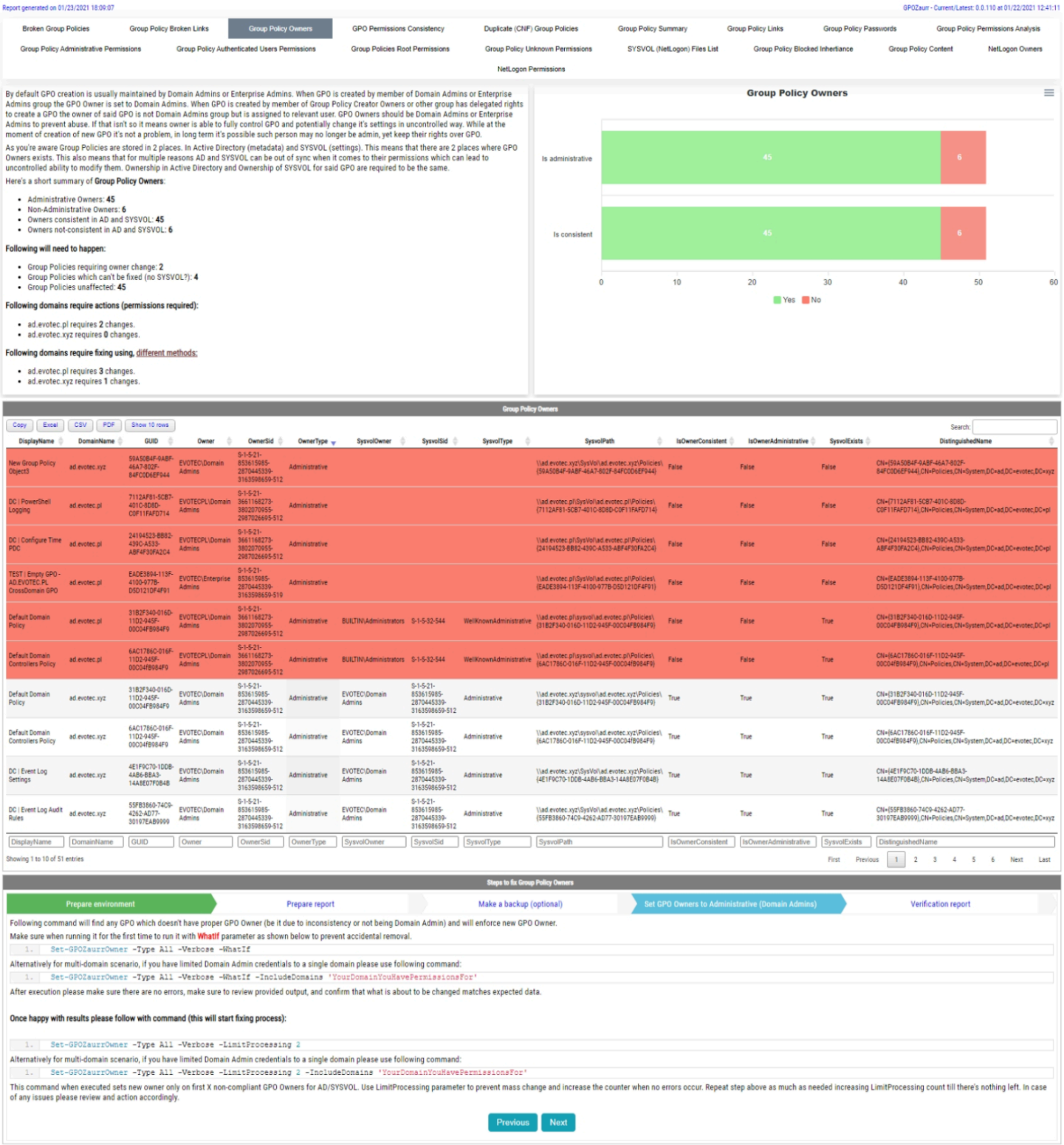


Invoke-GPOZaurr - Report GPOOwners

By default, GPO creation is usually maintained by Domain Admins or Enterprise Admins. When GPO is created by Domain Admins or Enterprise Admins group members, the GPO Owner is set to Domain Admins. When GPO is created by a member of Group Policy Creator Owners or other group has delegated rights to create a GPO, the owner of said GPO is not Domain Admins group but is assigned to the relevant user. GPO Owners should be Domain Admins or Enterprise Admins to prevent abuse. If that isn't so, it means the owner can fully control GPO and potentially change its settings in an uncontrolled way. While at the moment of creation of new GPO, it's not a problem, in the long term, it's possible such a person may no longer be admin, yet keep their rights over GPO. As your aware, Group Policies are stored in 2 places. In Active Directory (metadata) and SYSVOL (settings). This means that there are 2 places where GPO Owners exists. This also means that for multiple reasons, AD and SYSVOL can be out of sync when it comes to their permissions, which can lead to uncontrolled ability to modify them. Ownership in Active Directory and Ownership of SYSVOL for said GPO is required to be the same.

1. Invoke-GPOZaurr -Type GPOOwners

Tech Savvy Productions



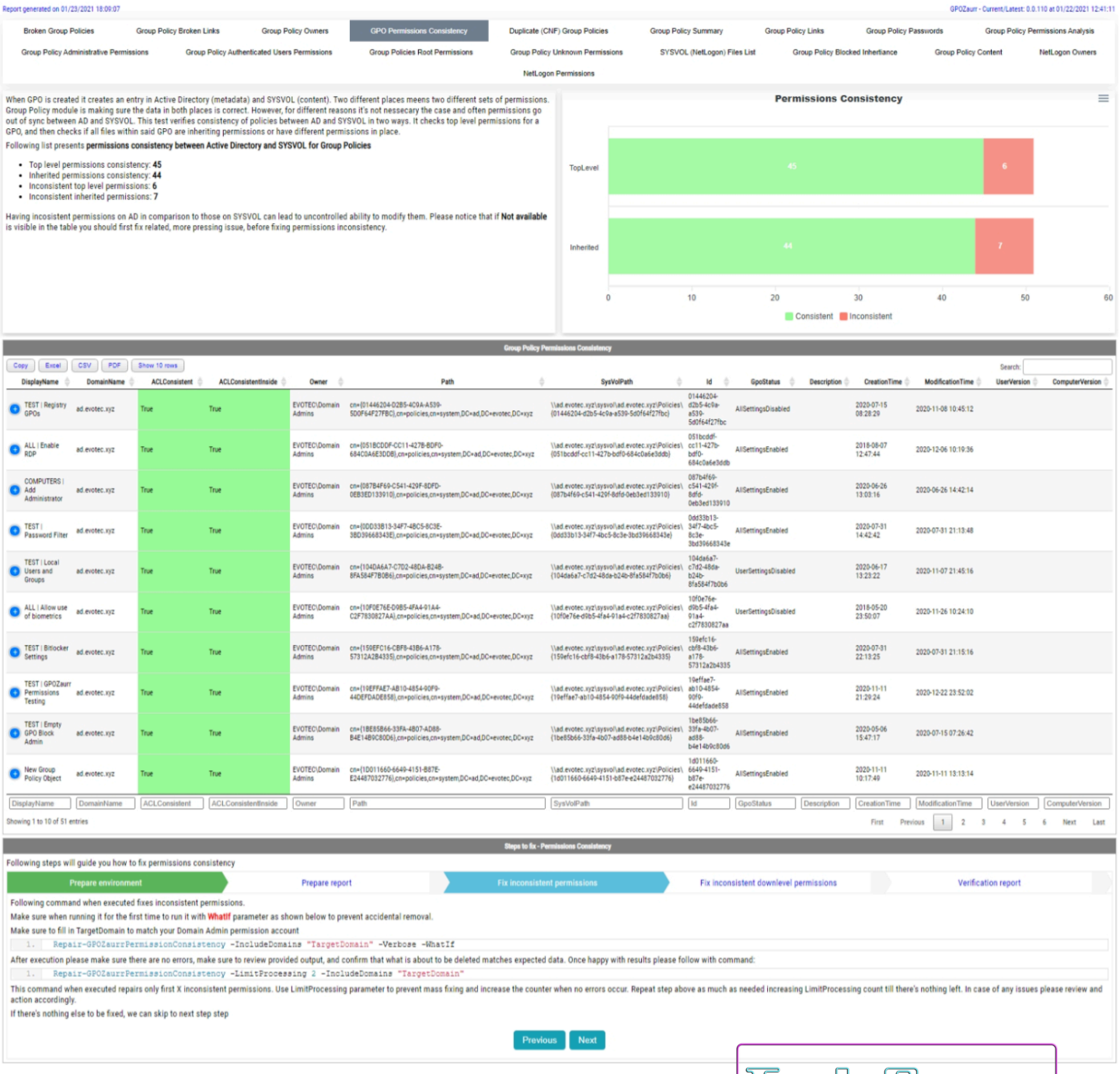
This report is fairly complete with detection and automated fix.

Invoke-GPOZaurr - Report GPOConsistency

Tech Savvy Productions

When GPO is created, it creates an entry in Active Directory (metadata) and SYSVOL (content). Two different places mean two different sets of permissions. The group Policy module is making sure the data in both places is correct. However, it's not necessarily the case for different reasons, and often permissions go out of sync between AD and SYSVOL. This test verifies the consistency of policies between AD and SYSVOL in two ways. It checks top-level permissions for a GPO and then checks if all files within said GPO is inheriting permissions or have different permissions in place.

1. Invoke-GPOZaurr -Type GPOConsistency



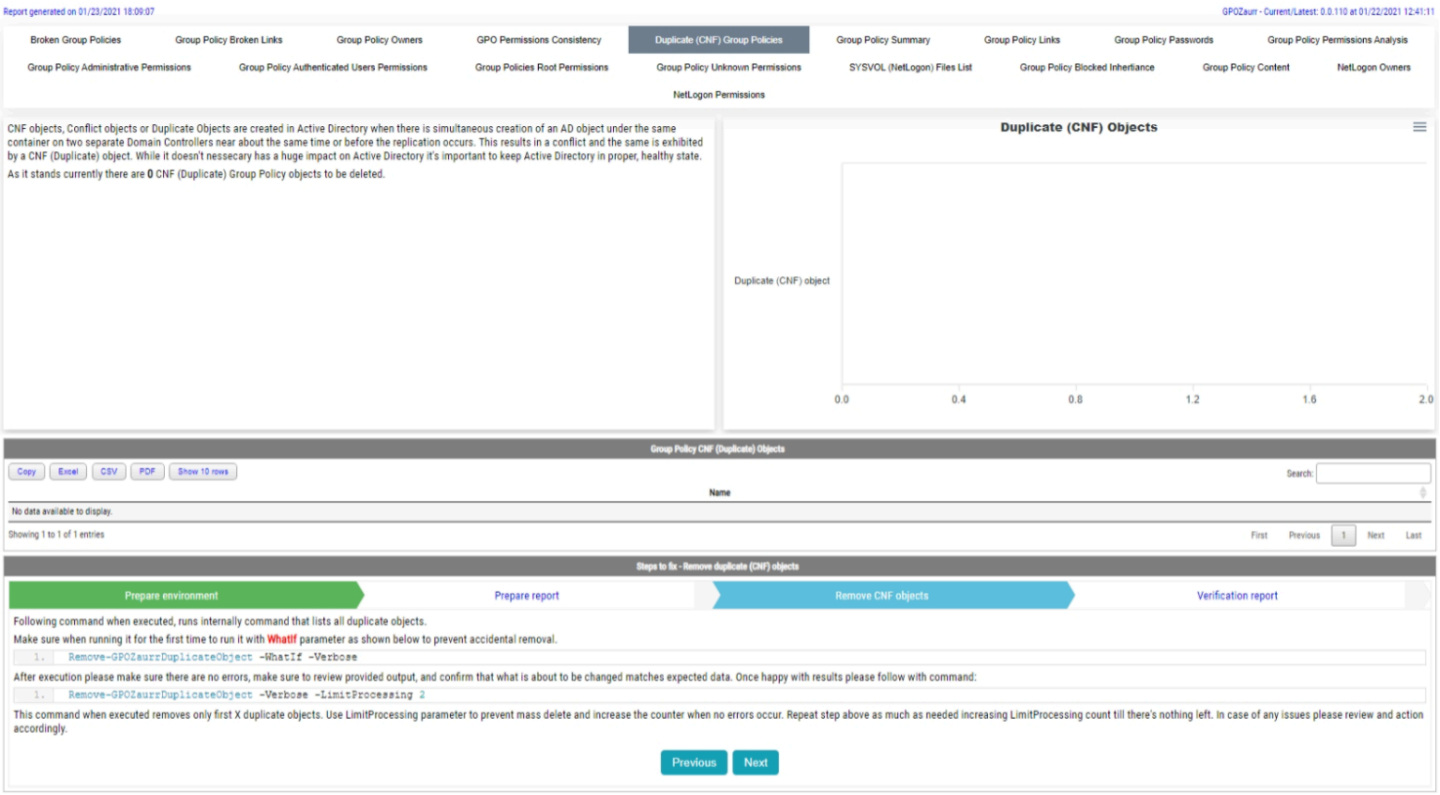
Tech Savvy Productions

This report is fairly complete and with an automated fix.

Invoke-GPOZaurr - Report GPODuplicates

CNF objects, Conflict objects, or Duplicate Objects are created in Active Directory when there is simultaneous creation of an AD object under the same container on two separate Domain Controllers near about the same time or before the replication occurs. This results in a conflict and a CNF (Duplicate) object exhibits the same. While it doesn't necessarily have a huge impact on Active Directory, it's important to keep Active Directory in a proper, healthy state.

1. Invoke-GPOZaurr -Type GPODuplicates



This report is fairly complete and with an automated fix. Be advised above screenshot doesn't show any detected problems because it's pretty hard to generate duplicated objects on-demand, so my test environment doesn't have any. But it does detect those.

Invoke-GPOZaurr - Report GPOList

Over time Administrators add more and more group policies as business requirements change. Due to neglect or thinking it may serve its purpose, later on, many Group Policies often have no value at all. Either the Group Policy is not linked to anything and stays unlinked forever, or GPO is linked, but the link (links) are disabled, or GPO is totally disabled. Then there are Group Policies that are targetting certain groups or persons, and that group is removed, leaving

Tech Savvy Productions

Group Policy doing nothing. Additionally, sometimes new GPO is created without any settings, or the settings are removed over time, but GPO stays in place.

1. Invoke-GPOZaurr -Type GPOList

“Perhaps it is no wonder that the women were first at the Cradle and last at the Cross. They had never known a man like this Man - there never has been such another. A prophet and teacher who never nagged at them, never flattered or coaxed or patronized; who never made arch jokes about them, never treated them either as "The women, God help us!" or "The ladies, God bless them!"; who rebuked without querulousness and praised without condescension; who took their questions and arguments seriously; who never mapped out their sphere for them, never urged them to be feminine or jeered at them for being female; who had no axe to grind and no uneasy male dignity to defend; who took them as he found them and was completely unself-conscious. There is no act, no sermon, no parable in the whole Gospel that borrows its pungency from female perversity; nobody could possibly guess from the words and deeds of Jesus that there was anything "funny" about woman's nature.”

— Dorothy L. Sayers

Tech Savvy Productions

Report generated on 01/23/2021 18:09:07

GPOZaurr - Current/Latest: 0.0.110 at 01/22/2021 12:41:11

Broken Group Policies | Group Policy Broken Links | Group Policy Owners | GPO Permissions Consistency | Duplicate (CNF) Group Policies | **Group Policy Summary** | Group Policy Links | Group Policy Passwords | Group Policy Permissions Analysis | Group Policy Administrative Permissions | Group Policy Authenticated Users Permissions | Group Policies Root Permissions | Group Policy Unknown Permissions | SYSVOL (NetLogon) Files List | Group Policy Blocked Inheritance | Group Policy Content | NetLogon Owners | NetLogon Permissions

Over time Administrators add more and more group policies, as business requirements change. Due to neglection or thinking it may serve it's purpose later on a lot of Group Policies often have no value at all. Either the Group Policy is not linked to anything and just stays unlinked forever, or GPO is linked, but the link (links) are disabled or GPO is totally disabled. Then there are Group Policies that are targetting certain group or person and that group is removed leaving Group Policy doing nothing. Additionally sometimes new GPO is created without any settings or the settings are removed over time, but GPO stays in place.

- Group Policies total: **51**
- Group Policies valid: **14**
- Group Policies **NOT** valid: **37**
  - Group Policies that are unlinked (are not doing anything currently): **30**
  - Group Policies that are empty (have no settings): **17**
  - Group Policies that are linked, but empty: **5**
  - Group Policies that are linked, but link disabled: **1**
  - Group Policies that are disabled (both user/computer sections): **2**
  - Group Policies that have no Apply Permission: **2**
- Group Policies **NOT** valid, to skip: **0** (not older than 7 days)
- Group Policies younger than 7 days: **0** (not older than 7 days)

Following domains require actions (permissions required):

- ad.evotec.pl requires **3** changes.
- ad.evotec.xyz requires **34** changes.

Keep in mind that each GPO can match multiple conditions such as being empty and unlinked and disabled at the same time. We're only deleting GPO once.

All **empty** or **unlinked** or **disabled** Group Policies can be automatically deleted. Please review output in the table and follow steps below table to cleanup Group Policies. GPOs that have content, but are disabled require manual intervention. If performance is an issue you should consider disabling user or computer sections of GPO when those are not used.

Additionally, we're reviewing Group Policies that have their section disabled, but contain data.

- Group Policies with problems: **3**
  - Group Policies that have content (computer), but are disabled: **2**
  - Group Policies that have content (user), but are disabled: **1**

Such policies require manual review from whoever owns them. It could be a mistake tha section was disabled while containing data or that content is no longer needed in which case it should be deleted. This can't be auto-handled and is INFORMATIONAL only.

Following domains require actions (permissions required):

- ad.evotec.pl requires **0** changes.
- ad.evotec.xyz requires **3** changes.

Moreover, for best performance it's recommended that if there are no settings of certain kind (Computer or User settings) it's best to disable whole section.

- Group Policies with optimization:
  - Group Policies that are optimized (computer) **26**
  - Group Policies that are optimized (user): **14**
- Group Policies without optimization:
  - Group Policies that are not optimized (computer): **25**
  - Group Policies that are not optimized (user): **37**

This means **47** could be optimized for performance reasons.

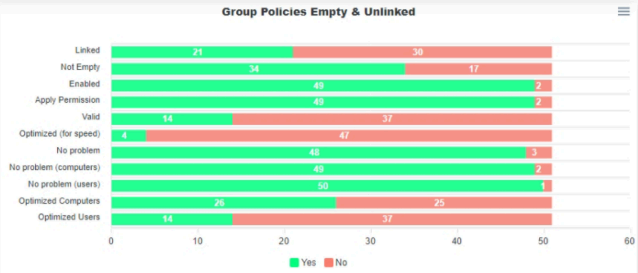
Following domains require actions (permissions required):

- ad.evotec.pl requires **3** changes.
- ad.evotec.xyz requires **44** changes.

To generate up to date report please execute:

- Install-Module GPOZaurr -Force or install module manually.
- Invoke-GPOZaurr -FilePath $Env:UserProfile\Desktop\GPOZaurrGPOPListBefore.html -Verbose -Type GPOList

Steps above will generate above summary with more details allowing you to get up to date report and steps on how to fix it.



Group Policies List

Explanation to table columns:

- **Empty** - means GPO has currently no content. It could be there was content, but it was removed, or that it never had content.
- **Linked** - means GPO is linked or unlinked. We need at least one link that is enabled to mark it as linked. If GPO is linked, but all links are disabled, it's not linked.
- **Enabled** - means GPO has at least one section enabled. If enabled is set to false that means both sections are disabled, and therefore GPO is not active.
- **Optimized** - means GPO section that is not in use is disabled. If section (user or computer) is enabled and there is no content, it's not optimized.
- **Problem** - means GPO has one or more section (user or computer) that is disabled, yet there is content in it.
- **ApplyPermission** - means GPO has no Apply Permission. This means there's no user/computer/group it's applicable to.

Copy | Excel | CSV | PDF | Show 10 rows — Search:

DisplayName	DomainName	GUID	Days	Empty	Linked	Enabled	Optimized	Problem	ApplyPermission	Exclude	ComputerPolicies	UserPolicies	LinksCount	LinksEnabledCount	LinksDisabledCount	EnabledDetails	ComputerProblem	ComputerOptimized
TEST | Registry GPOs	ad.evotec.xyz	01446204-d2b5-4c9a-a539-5d0f64f27fbc	76	False	False	False	False	True	True	False	Windows Registry		0	0		All settings disabled	True	False
ALL | Enable RDP	ad.evotec.xyz	051bcdef-cc11-427b-bdf0-684c0a6e3ddb	48	False	True	True	False	False	True	False	Registry		7	6	1	Enabled	False	True
COMPUTERS | Add Administrator	ad.evotec.xyz	087b4f60-c541-429f-8dfe-0eb3ed133910	211	False	False	True	False	False	True	False	Local Users and Groups		0	0	0	Enabled	False	True
TEST | Password Filter	ad.evotec.xyz	0dd33b13-34f7-4bc5-8c3e-3bd39666834e	175	False	False	True	False	False	True	False	Registry		0	0	0	Enabled	False	True
TEST | Local Users and Groups	ad.evotec.xyz	104da6a7-c7d2-48de-b24b-8fa584f7b0b6	76	False	False	True	False	True	True	False	Local Users and Groups, Services	Local Users and Groups	0	0	0	User configuration settings disabled	False	True
ALL | Allow use of biometrics	ad.evotec.xyz	10f0e76e-d0b5-4fa4-91a4-c2f7830827aa	58	False	True	True	True	False	True	False	Name Resolution Policy, Registry		4	3	1	User configuration settings disabled	False	True
TEST | Bitlocker Settings	ad.evotec.xyz	159efc16-cbf8-43b6-a17b-57312a2b4335	175	False	False	True	False	False	True	False	Registry		0	0	0	Enabled	False	True
TEST | GPOZaurr Permissions Testing	ad.evotec.xyz	19effae7-ab10-4854-90f9-44defdade858	31	False	True	True	False	False	True	False		Registry	1	1	0	Enabled	False	False
TEST | Empty GPO Block Admin	ad.evotec.xyz	1be85b66-85fe-4b07-ad88-b4e14b9c80d6	192	True	False	True	False	False	True	False			0	0	0	Enabled	False	False
New Group Policy Object	ad.evotec.xyz	1d011660-6649-4151-b87e-e24487032776	73	True	False	True	False	False	False	False			0	0	0	Enabled	False	False

Showing 1 to 10 of 51 entries — First | Previous | 1 | 2 | 3 | 4 | 5 | 6 | Next | Last

Steps to fix - Empty & Unlinked & Disabled Group Policies

Following steps will guide you how to remove empty or unlinked group policies

Prepare environment | Prepare report | Make a backup | Excluding Group Policies | Remove GPOs that are EMPTY | Remove GPOs that are UNLINKED | Remove GPOs that are DISABLED | Remove GPOs that do not APPLY | Optimize GPOs (optional) | Verification report

Following command when executed removes every **EMPTY** Group Policy. Make sure when running it for the first time to run it with **Whatif** parameter as shown below to prevent accidental removal.**Make sure to use BackupPath which will make sure that for each GPO that is about to be deleted a backup is made to folder on a desktop.**You can skip parameters related to backup if you did backup all GPOs prior to running remove command.

```
1.  Remove-GPOZaurr -RequireDays 7 -Type Empty -BackupPath "$Env:UserProfile\Desktop\GPO" -Verbose -WhatIf
```

Alternatively for multi-domain scenario, if you have limited Domain Admin credentials to a single domain please use following command:

```
1.  Remove-GPOZaurr -RequireDays 7 -Type Empty -BackupPath "$Env:UserProfile\Desktop\GPO" -Verbose -WhatIf -IncludeDomains 'YourDomainYouHavePermissionsFor'
```

After execution please make sure there are no errors, make sure to review provided output, and confirm that what is about to be deleted matches expected data.

Once happy with results please follow with command (this will start fixing process):

```
1.  Remove-GPOZaurr -RequireDays 7 -Type Empty -BackupPath "$Env:UserProfile\Desktop\GPO" -LimitProcessing 2 -Verbose
```

Alternatively for multi-domain scenario, if you have limited Domain Admin credentials to a single domain please use following command:

```
1.  Remove-GPOZaurr -RequireDays 7 -Type Empty -BackupPath "$Env:UserProfile\Desktop\GPO" -LimitProcessing 2 -Verbose -IncludeDomains 'YourDomainYouHavePermissionsFor'
```

This command when executed deletes only first X empty GPOs. Use LimitProcessing parameter to prevent mass delete and increase the counter when no errors occur.Repeat step above as much as needed increasing LimitProcessing count till there's nothing left. In case of any issues please study and action accordingly.Please make sure to check if backup is made as well before going all in.

If there's nothing else to be deleted, we can skip to next step step

Previous | Next

Tech Savvy Productions

This report is fairly complete and provides automated fixes for most issues detected.

Invoke-GPOZaurr - Report GPOPermissions

The following report contains a full overview of all permissions around Group Policies. It detects 4 different problems (lack of authenticated users, wrong permissions for Domain Admins and Enterprise Admins, Unknown permissions, and lack of proper permission for SYSTEM account). It also contains all permissions, so it's easy to review all permissions from a single place. For each problem, automation is developed, so it's fairly easy to fix any issues as long as you agree with what's proposed.

1. Invoke-GPOZaurr -Type GPOPermissions

“The great advantage about telling the truth is that nobody ever believes it.”

— Dorothy L. Sayers

Tech Savvy Productions

Report generated on 01/24/2021 11:27:36

GPOZaurr - Current/Latest: 0.0.110 at 01/22/2021 12:41:11

When GPO is created it gets a handful of standard permissions. Those are:

- NT AUTHORITY\Authenticated Users with GpoApply permissions
- Domain Admins and Enterprise Admins with Edit/Delete/Modify permissions
- SYSTEM account with Edit/Delete/Modify permissions

But then IT people change those permissions to their own needs. While most changes make sense and are required to be able to target proper groups of people, some changes are not required or even bad.

First problem relates to NT AUTHORITY\Authenticated Users

When GPO is created one of the permissions that are required for proper functioning of Group Policies is NT AUTHORITY\Authenticated Users. Some Administrators don't follow best practices and trying to remove GpoApply permission, remove also GpoRead permission from a GPO which can have consequences. On June 14th, 2016 Microsoft released MoERa that requires Authenticated Users to be present on all Group Policies to function properly. MS16-072 changes the security context with which user group policies are retrieved. This by-design behavior change protects customers' computers from a security vulnerability.

- Before MS16-072 is installed, user group policies were retrieved by using the user's security context.
- After MS16-072 is installed, user group policies are retrieved by using the computer's security context.

Unfortunetly it's not as simple as it sounds. While checking for permissions mostly works fine, it's possible a Group Policy has totally removed account permissions from being able to asses any of it. The account we're using EVOTEC\PRZEMYSLAW.KLYS may simply not have enough permisions to properly asses permissions for a GPO. Therefore we're using dual assesment for this situation, where first assesment is checking for GPO visibility or lack of it, and second assesment is checking for direct permissions assignement. We just were able to detect the problem, but hopefully higher level account (Domain Admin) should be able to provide full assesment.

First assesment results:

- Group Policies couldn't read at all: **1**
- Group Policies with permissions allowing read: **51**

Following domains require actions (permissions required):

- ad.evotec.pl requires **0** changes out of 5.
- ad.evotec.xyz requires **1** changes out of 47.

Second Assesment results

- Group Policies requiring Authenticated Users with GpoRead permission: **2**
- Group Policies which don't require changes: **49**

Following domains require actions (permissions required):

- ad.evotec.pl requires **0** changes.
- ad.evotec.xyz requires **2** changes.

Second problem relates to Domain Admins and Enterprise Admins

When GPO is created by default it gets Domain Admins and Enterprise Admins with Edit/Delete/Modify Security permissions. For some reason, some Administrators remove those permissions or modify them when they shouldn't touch those at all. Since having Edit/Delete/Modify Security permissions doesn't affect GPOApply permissions there's no reason to remove Domain Admins or Enterprise Admins from permissions, or limit their rights. Domain Admins and Enterprise Admins have to have either GPOEditModify permissions or at the very least GPOCustom. When GPOCustom is set it usually means there's a mix of Allow and Deny permission in place (for example deny GPOApply). In such case we're assuming you know what you're doing. However it's always possible to review those permissions, as those are marked in the table for review.

Assesment results

- Group Policies requiring Administrative permission fix: **2**
- Group Policies which don't require changes: **49**

Following domains require actions (permissions required):

- ad.evotec.pl requires **1** changes.
- ad.evotec.xyz requires **1** changes.

Third problem relates to SYSTEM account

When GPO is created by default it gets SYSTEM account with Edit/Delete/Modify Security permissions. For some reason, some Administrators remove those permissions or modify them when they shouldn't touch those at all. Since having Edit/Delete/Modify Security permissions doesn't affect GPOApply permissions there's no reason to remove SYSTEM from permissions, or limit their rights.

Assesment results

- Group Policies requiring SYSTEM permission fix: **1**
- Group Policies which don't require changes: **50**

Following domains require actions (permissions required):

- ad.evotec.pl requires **0** changes.
- ad.evotec.xyz requires **1** changes.

Fourth problem relates to UNKNOWN SID

Sometimes groups or users are deleted in Active Directory and unfortunetly their permissions are not cleaned automatically. Those are left in-place and stay there forever until removed.

Assesment results

- Group Policies requiring Unknown permission removal: **1**
- Group Policies which don't require changes: **50**

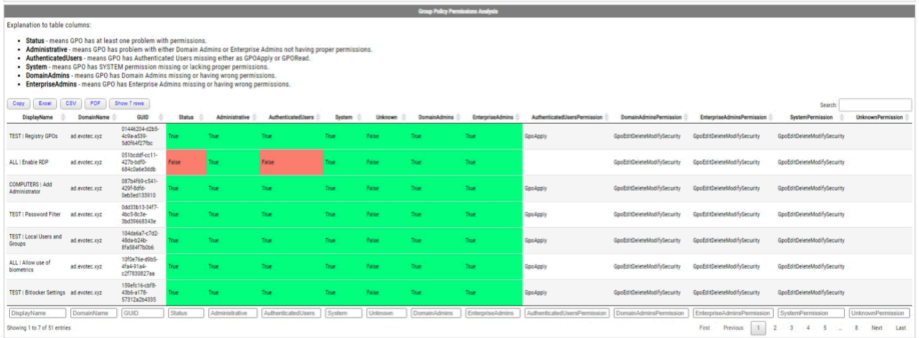
Following domains require actions (permissions required):

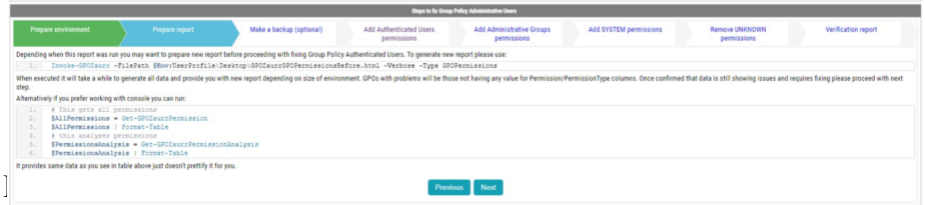
- ad.evotec.pl requires **0** changes.
- ad.evotec.xyz requires **1** changes.

To generate up to date report please execute:

- Install-Module GPOZaurr -Force or install module manually.
- Invoke-GPOZaurr -FilePath $Env:UserProfile\Desktop\GPOZaurr\GPOPermissionsBefore.html -Verbose -Type GPOPermissions

Steps above will generate above summary with more details allowing you to get up to date report and steps on how to fix it.





Tech Savvy Productions

This report is interactive, meaning clicking on a GPO in one table limits permissions shown in another table. **GPOPermissions** type is kind of ultimate way for you to deal with permissions. I've made one report that covers what 3 different reports were covering before.

1. Invoke-GPOZaurr -Type GPOPermissionsRead,GPOPermissionsAdministrative,GPOPermissionsUnknown

So while you can use the cmdlet above with each type separately – it's easier to use one.

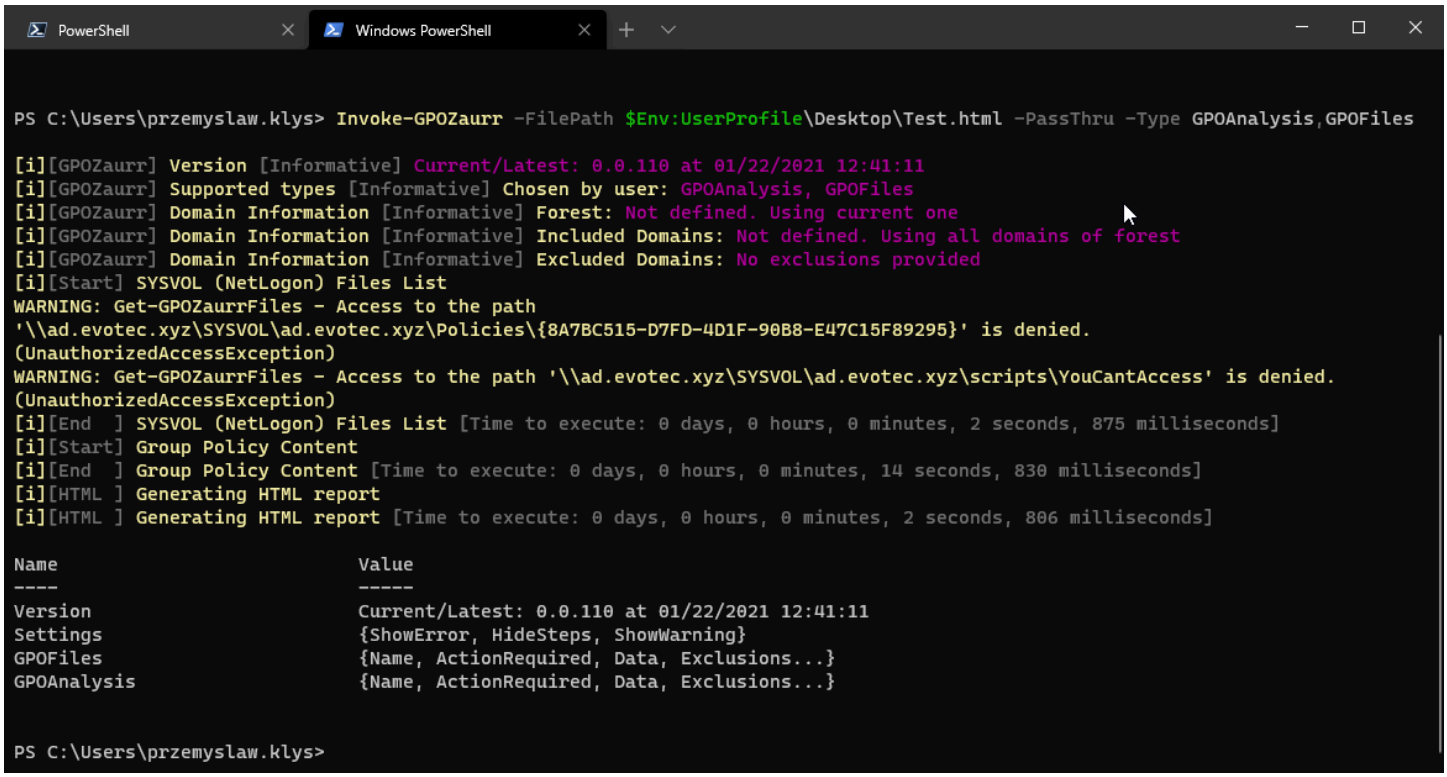
Invoke-GPOZaurr - advanced usage

Invoke-GPOZaurr is basically a wrapper of around 20 or so different GPO cmdlets that I have developed over a period of six months. I was worried that with so many cmdlets being available in my module and my laziness in the documentation, I thought **Invoke-GPOZaurr's** three-step approach (Describe Problem, Provide Data, Offer Solution) was an experiment that I believe will help me manage my GPOs efficiently for years to come. Not everything is completed, but at the current state, it's good enough for release. It allows you to understand where you stand without spending days, weeks, or months of analysis depending on how big your Active Directory is. Of course, this one little command has few more options that allow for different customization options.

1. Invoke-GPOZaurr [[-Type] <string[]>] [[-ExcludeGroupPolicies] <scriptblock>] [-FilePath <string>] [-PassThru] [-HideHTML] [-HideSteps] [-ShowError] [-ShowWarning] [-Forest <string>] [-ExcludeDomains <string[]>] [-IncludeDomains <string[]>] [<CommonParameters>]

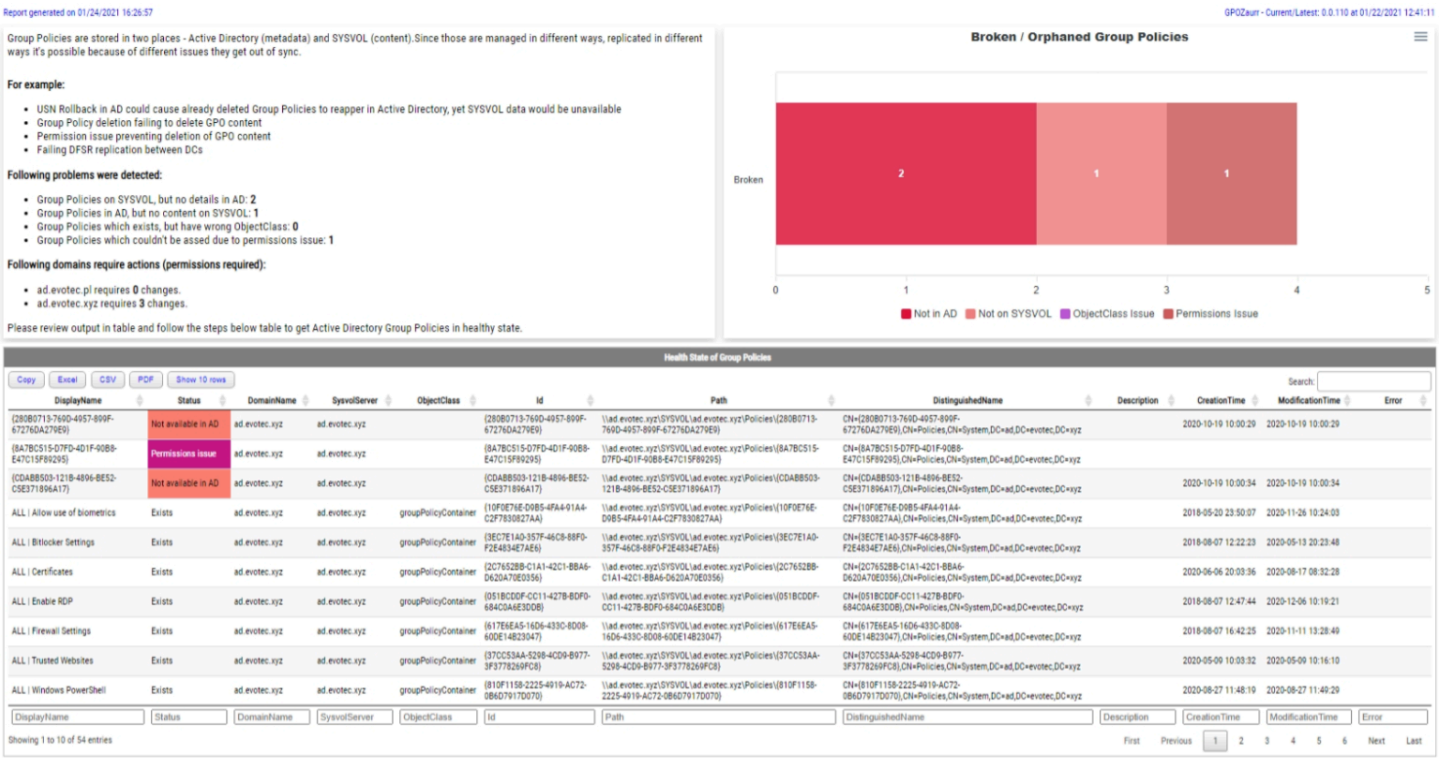
Using a **Type** parameter, you can ask for one or multiple types. Providing **FilePath** parameter, you can tell **GPOZaurr** where to save created **HTML** file. PassThru, on the other hand, is useful to have **HTML** generated and get the output of the reports back to you for future analysis.

Tech Savvy Productions



It's also possible to hide steps to fix a given problem. This can be useful if you're doing an overview for your Client/Management and don't want to show how to fix it.

1. Invoke-GPOZaurr -FilePath $Env:UserProfile\Desktop\Test.html -Type GPOBroken -HideSteps



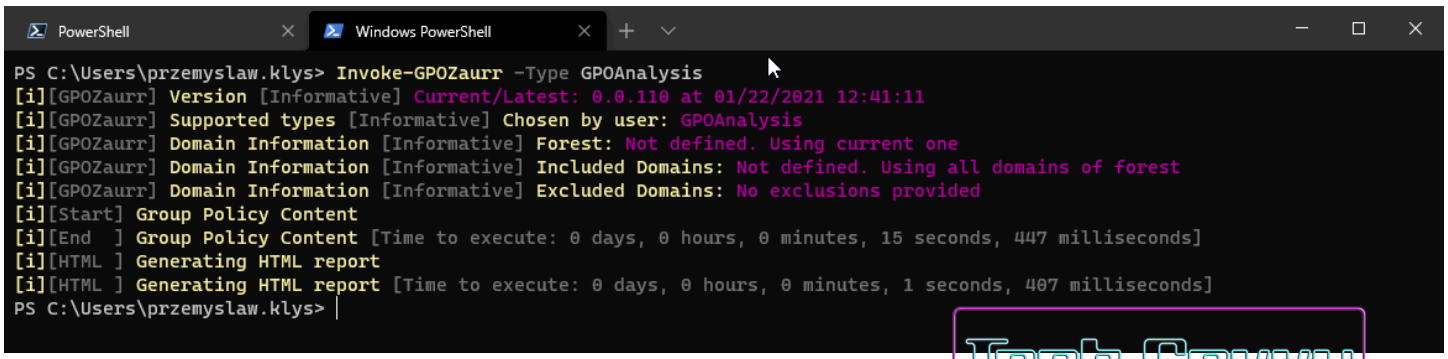
Using **HideHTML** parameter prevents auto-opening of HTML. It's useful for automation purposes.

1. Invoke-GPOZaurr -FilePath $Env:UserProfile\Desktop\Test.html -Type GPOBroken -HideSteps -HideHTML

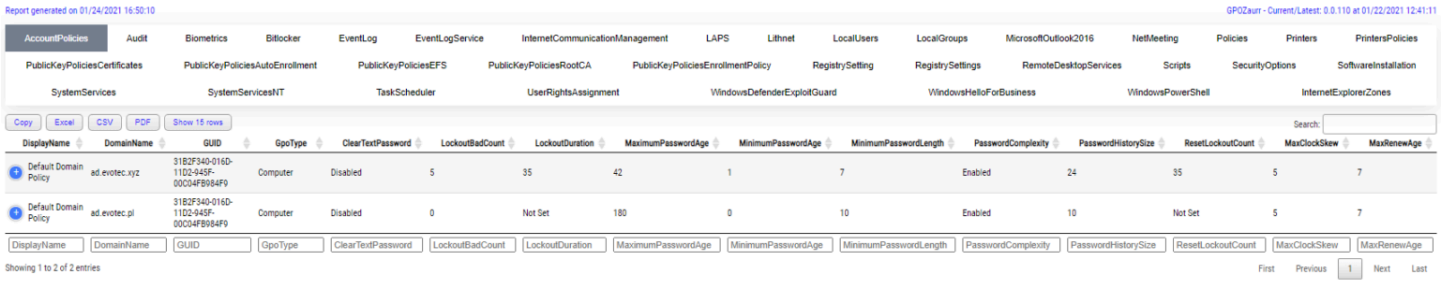
Invoke-GPOZaurr - Type GPOAnalysis

GPO Analysis report is one of the coolest ones I've made. It's able to provide a lot of smaller reports that show the content of group policies. Each report is a separate tab. Using **GPO GUI**, you would normally show you similar output, but this one does it globally. If you've ever tried to find all GPOs that map drives, find ones that have script execution – it's the way to go.

1. Invoke-GPOZaurr -Type GPOAnalysis



Tech Savvy Productions



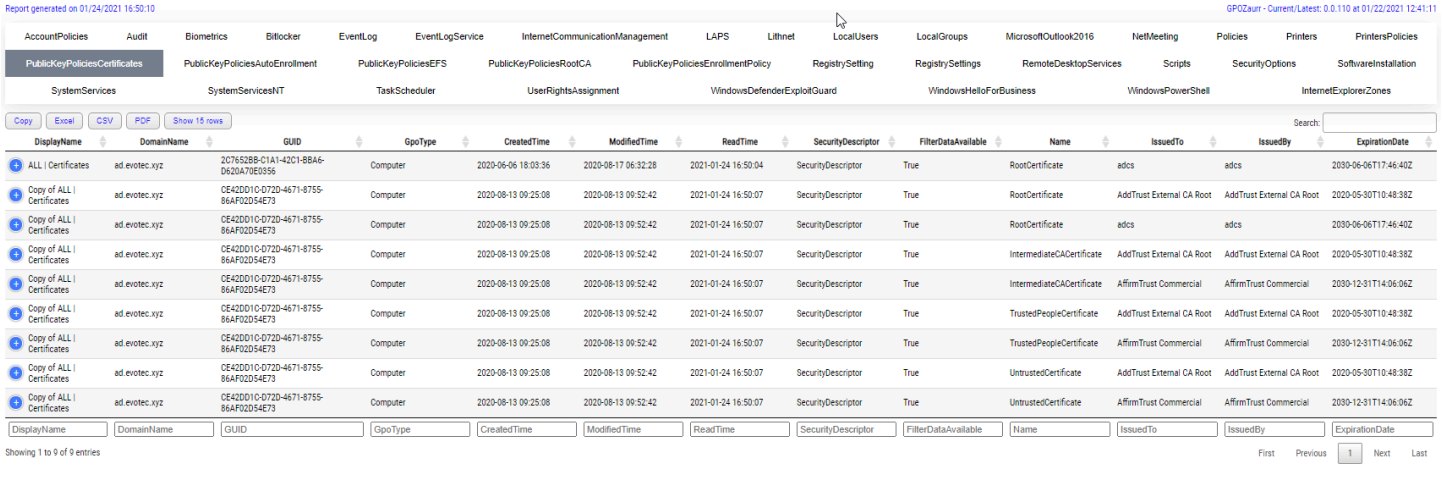
Report generated on 01/24/2021 16:50:10 — GPOZaurr - Current/Latest: 0.0.110 at 01/22/2021 12:41:11

AccountPolicies | Audit | Biometrics | Bitlocker | EventLog | EventLogService | InternetCommunicationManagement | LAPS | Lithnet | LocalUsers | LocalGroups | MicrosoftOutlook2016 | NetMeeting | Policies | Printers | PrintersPolicies
PublicKeyPoliciesCertificates | PublicKeyPoliciesAutoEnrollment | PublicKeyPoliciesEFS | PublicKeyPoliciesRootCA | PublicKeyPoliciesEnrollmentPolicy | RegistrySetting | RegistrySettings | RemoteDesktopServices | Scripts | SecurityOptions | SoftwareInstallation
SystemServices | SystemServicesNT | TaskScheduler | UserRightsAssignment | WindowsDefenderExploitGuard | WindowsHelloForBusiness | WindowsPowerShell | InternetExplorerZones

Copy | Excel | CSV | PDF | Show 15 rows — Search:

DisplayName	DomainName	GUID	GpoType	ClearTextPassword	LockoutBadCount	LockoutDuration	MaximumPasswordAge	MinimumPasswordAge	MinimumPasswordLength	PasswordComplexity	PasswordHistorySize	ResetLockoutCount	MaxClockSkew	MaxRenewAge
Default Domain Policy	ad.evotec.xyz	31B2F340-016D-11D2-945F-00C04FB984F9	Computer	Disabled	5	35	42	1	7	Enabled	24	35	5	7
Default Domain Policy	ad.evotec.pl	31B2F340-016D-11D2-945F-00C04FB984F9	Computer	Disabled	0	Not Set	180	0	10	Enabled	10	Not Set	5	7

Showing 1 to 2 of 2 entries — First | Previous | 1 | Next | Last



Report generated on 01/24/2021 16:50:10 — GPOZaurr - Current/Latest: 0.0.110 at 01/22/2021 12:41:11

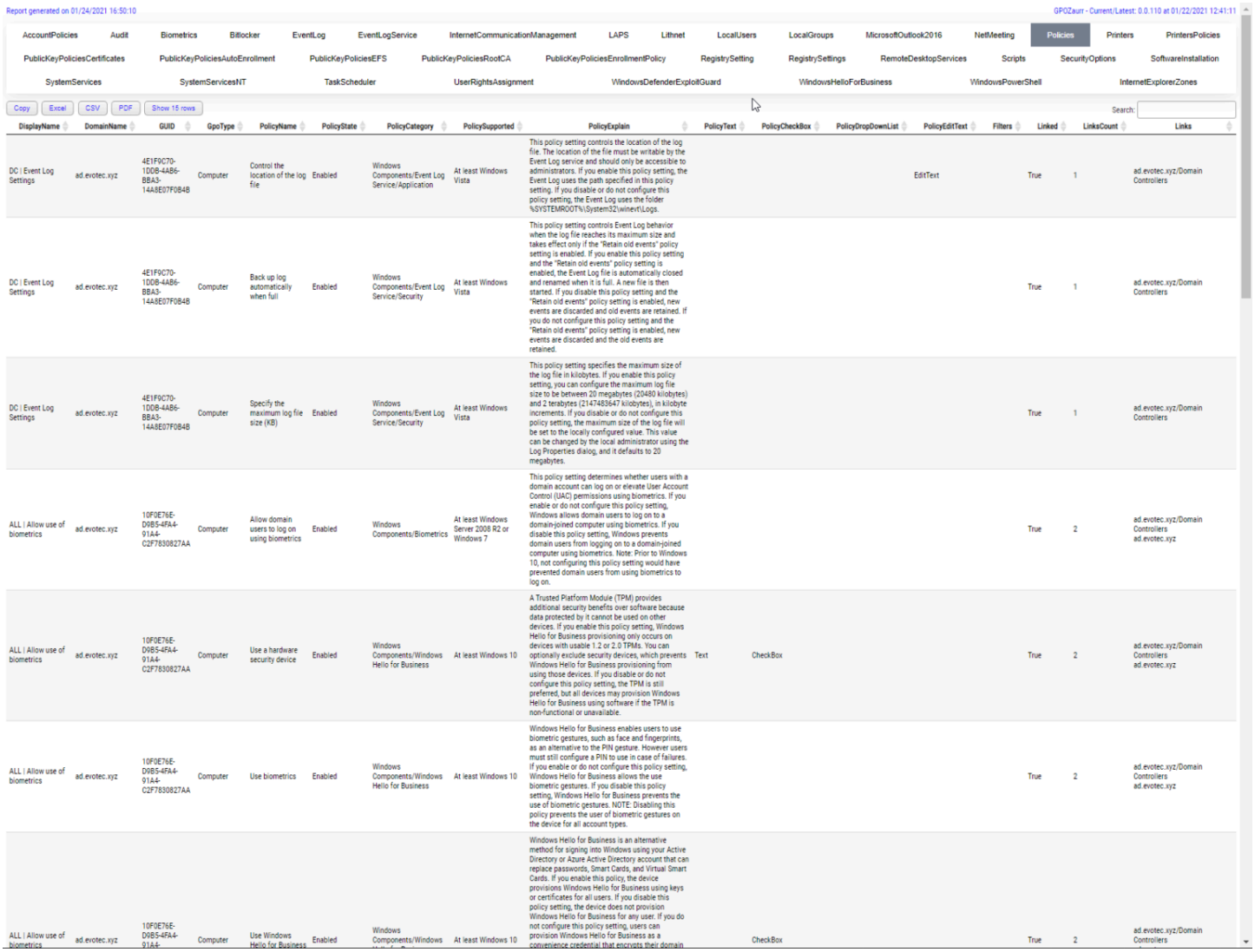
AccountPolicies | Audit | Biometrics | Bitlocker | EventLog | EventLogService | InternetCommunicationManagement | LAPS | Lithnet | LocalUsers | LocalGroups | MicrosoftOutlook2016 | NetMeeting | Policies | Printers | PrintersPolicies
PublicKeyPoliciesCertificates | PublicKeyPoliciesAutoEnrollment | PublicKeyPoliciesEFS | PublicKeyPoliciesRootCA | PublicKeyPoliciesEnrollmentPolicy | RegistrySetting | RegistrySettings | RemoteDesktopServices | Scripts | SecurityOptions | SoftwareInstallation
SystemServices | SystemServicesNT | TaskScheduler | UserRightsAssignment | WindowsDefenderExploitGuard | WindowsHelloForBusiness | WindowsPowerShell | InternetExplorerZones

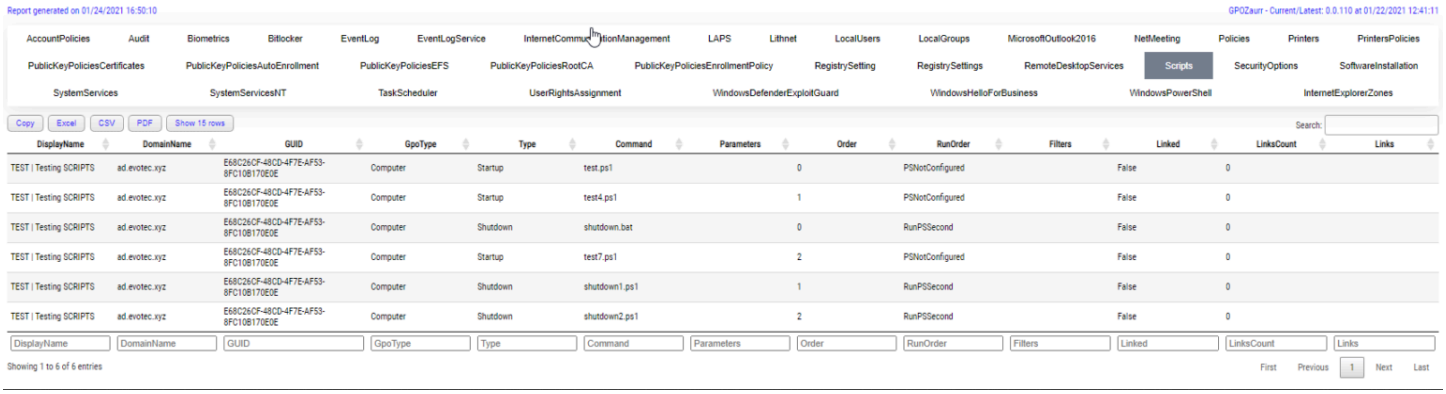
Copy | Excel | CSV | PDF | Show 15 rows — Search:

DisplayName	DomainName	GUID	GpoType	CreatedTime	ModifiedTime	ReadTime	SecurityDescriptor	FilterDataAvailable	Name	IssuedTo	IssuedBy	ExpirationDate
ALL | Certificates	ad.evotec.xyz	2C7652BB-C1A1-42C1-BBA6-D620A70E0356	Computer	2020-06-06 18:03:36	2020-08-17 06:32:28	2021-01-24 16:50:04	SecurityDescriptor	True	RootCertificate	adcs	adcs	2030-06-06T17:46:40Z
Copy of ALL | Certificates	ad.evotec.xyz	CE42DD1C-D72D-4671-8755-86AF02D54E73	Computer	2020-08-13 09:25:08	2020-08-13 09:52:42	2021-01-24 16:50:07	SecurityDescriptor	True	RootCertificate	AddTrust External CA Root	AddTrust External CA Root	2020-05-30T10:48:38Z
Copy of ALL | Certificates	ad.evotec.xyz	CE42DD1C-D72D-4671-8755-86AF02D54E73	Computer	2020-08-13 09:25:08	2020-08-13 09:52:42	2021-01-24 16:50:07	SecurityDescriptor	True	RootCertificate	adcs	adcs	2030-06-06T17:46:40Z
Copy of ALL | Certificates	ad.evotec.xyz	CE42DD1C-D72D-4671-8755-86AF02D54E73	Computer	2020-08-13 09:25:08	2020-08-13 09:52:42	2021-01-24 16:50:07	SecurityDescriptor	True	IntermediateCACertificate	AddTrust External CA Root	AddTrust External CA Root	2020-05-30T10:48:38Z
Copy of ALL | Certificates	ad.evotec.xyz	CE42DD1C-D72D-4671-8755-86AF02D54E73	Computer	2020-08-13 09:25:08	2020-08-13 09:52:42	2021-01-24 16:50:07	SecurityDescriptor	True	IntermediateCACertificate	AffirmTrust Commercial	AffirmTrust Commercial	2030-12-31T14:06:06Z
Copy of ALL | Certificates	ad.evotec.xyz	CE42DD1C-D72D-4671-8755-86AF02D54E73	Computer	2020-08-13 09:25:08	2020-08-13 09:52:42	2021-01-24 16:50:07	SecurityDescriptor	True	TrustedPeopleCertificate	AddTrust External CA Root	AddTrust External CA Root	2020-05-30T10:48:38Z
Copy of ALL | Certificates	ad.evotec.xyz	CE42DD1C-D72D-4671-8755-86AF02D54E73	Computer	2020-08-13 09:25:08	2020-08-13 09:52:42	2021-01-24 16:50:07	SecurityDescriptor	True	TrustedPeopleCertificate	AffirmTrust Commercial	AffirmTrust Commercial	2030-12-31T14:06:06Z
Copy of ALL | Certificates	ad.evotec.xyz	CE42DD1C-D72D-4671-8755-86AF02D54E73	Computer	2020-08-13 09:25:08	2020-08-13 09:52:42	2021-01-24 16:50:07	SecurityDescriptor	True	UntrustedCertificate	AddTrust External CA Root	AddTrust External CA Root	2020-05-30T10:48:38Z
Copy of ALL | Certificates	ad.evotec.xyz	CE42DD1C-D72D-4671-8755-86AF02D54E73	Computer	2020-08-13 09:25:08	2020-08-13 09:52:42	2021-01-24 16:50:07	SecurityDescriptor	True	UntrustedCertificate	AffirmTrust Commercial	AffirmTrust Commercial	2030-12-31T14:06:06Z

Showing 1 to 9 of 9 entries — First | Previous | 1 | Next | Last

Tech Savvy Productions





Tech Savvy Productions

The idea for every report is that each setting is stored per each line. This sometimes means that if the setting has a potential of 50 options, the report will generate 50 columns. I've not found an easy way to make it readable without custom creating and every report. While I do that for some of the reports, some are totally autogenerated. If you feel something is not covered or require a better report, open up an issue, and we can see what can be done.

Invoke-GPOZaurr - Automating GPOZaurr to Email

Since I want to keep my group policies healthy at all times, I've developed small automation. This automation deals with one report and sends an email to a ticketing system if there is a problem or sends an update to the AD team that everything is great. This automation uses PSWriteHTML (which is also used to generate **HTML** anyway). I've developed the module where the description on each report is available to use outside of **GPOZaurr** (that's where the **PassThru** parameter is useful).

```
Import-Module GPOZaurr -Force

$PasswordSecureString = 'passwordSecureString'

$Types = @(
@{
Name = 'GPOOwners'
Path = "$PSScriptRoot\Reports\GPOOwners_$(Get-Date -f yyyy-MM-dd_HHmmss).html"
Subject = '[AD Compliance] Group Policy Owners Issue'
Ticket = '[Ticket#2001000](https://linkToChangeRequest)'
Attach = $true
}
@{
Name = 'GPODuplicates'
Path = "$PSScriptRoot\Reports\GPODuplicates_$(Get-Date -f yyyy-MM-dd_HHmmss).html"
Subject = '[AD Compliance] Group Policy Duplicate (Conflicting) Objects Detected'
Ticket = '[Ticket#2001000](https://linkToChangeRequest)'
Attach = $true
}
@{
Name = 'NetLogonOwners'
Path = "$PSScriptRoot\Reports\NetLogonPermissions_$(Get-Date -f yyyy-MM-dd_HHmmss).html"
Subject = '[AD Compliance] NetLogon Owners Issue'
Ticket = '[Ticket#2001000](https://linkToChangeRequest)'
Attach = $true
}
@{
Name = 'GPOConsistency'
Path = "$PSScriptRoot\Reports\GPOConsistency_$(Get-Date -f yyyy-MM-dd_HHmmss).html"
Subject = '[AD Compliance] Group Policy Consistency'
Ticket = '[Ticket#2001000](https://linkToChangeRequest)'
Attach = $true
}
```


Tech Savvy Productions

```
# Too big
@{
Name = 'GPOPermissions'
Path = "$PSScriptRoot\Reports\GPOPermissions_$(Get-Date -f yyyy-MM-dd_HHmmss).html"
Subject = '[AD Compliance] Group Policy Permissions Analysis'
Ticket = '[Ticket#2001000](https://linkToChangeRequest)'
Attach = $false
}
@{
Name = 'GPOList'
Path = "$PSScriptRoot\Reports\GPOList_$(Get-Date -f yyyy-MM-dd_HHmmss).html"
Subject = '[AD Compliance] Group Policy Empty & Unlinked & Disabled Cleanup'
Ticket = '[Ticket#2001000](https://linkToChangeRequest)'
Attach = $false
}
@{
Name = 'GPOBroken';
Path = "$PSScriptRoot\Reports\GPOOrphans_$(Get-Date -f yyyy-MM-dd_HHmmss).html";
Subject = '[AD Compliance] Group Policy Orphaned/Broken Cleanup'
Ticket = '[Ticket#2001000](https://linkToChangeRequest)'
Attach = $true
}
@{
Name = 'GPOBrokenLink'
Path = "$PSScriptRoot\Reports\GPOBrokenLink_$(Get-Date -f yyyy-MM-dd_HHmmss).html"
Subject = '[AD Compliance] Group Policy Broken Links'
Ticket = '[Ticket#2001000](https://linkToChangeRequest)'
Attach = $true
}
)

foreach ($Type in $Types) {
$EmailHeaderBadReport = EmailHeader {
EmailFrom -Address 'EmailFrom@evotec.pl'
EmailTo -Addresses "przemyslawklys@evotec.pl", 'otherguy@evotec.pl'
EmailServer -Server 'smtpServer' -SSL -Port 25 -UserName 'login' -Password $PasswordSecureString
-PasswordAsSecure
EmailOptions -Priority High -DeliveryNotifications Never
EmailSubject -Subject $Type.Subject
if ($Type.Attach -eq $true) {
EmailAttachment -FilePath $Type.Path
}
}
$EmailHeaderGoodReport = EmailHeader {
EmailFrom -Address 'EmailFrom@evotec.pl'
EmailTo -Addresses "przemyslawklys@test.pl", 'otherguy@evotec.pl'
EmailServer -Server 'smtpServer' -SSL -Port 25 -UserName 'login' -Password $PasswordSecureString
-PasswordAsSecure
EmailOptions -Priority Low -DeliveryNotifications Never
EmailSubject -Subject $Type.Subject
if ($Type.Attach -eq $true) {
```


Tech Savvy Productions

```
EmailAttachment -FilePath $Type.Path
}
}
$ReportOutput = Invoke-GPOZaurr -FilePath $Type.Path -Type $Type.Name -PassThru -HideHTML -Verbose
foreach ($Report in $ReportOutput.Keys | Where-Object { $_ -notin 'Version', 'Settings' }) {
if ($ReportOutput[$Report]['ActionRequired'] -eq $true) {
Email {
$EmailHeaderBadReport
EmailBody {
EmailText -Text 'Hello Team,' -LineBreak
EmailText -Text "I've found disprepency in our domain that needs to be fixed and I need your help. "
-LineBreak

$ReportOutput[$Report]['Summary']

EmailText -LineBreak
EmailText -TextBlock {
"This automation was approved by CAB in $($Type.Ticket). "
"The goal is to keep Group Policies Healthy at all times! "
"In case of issues please contact Przemyslaw Klys "
} -LineBreak
EmailText -Text 'With regards,'
EmailText -Text 'Automated Monitoring'

} -FontSize 10pt
}
} else {
Email {
$EmailHeaderGoodReport
EmailBody {
EmailText -Text 'Hello Team,' -LineBreak
EmailText -Text "I've run the report and everything is looking great. Nothing to do here, but just wanted to say -
great job! " -LineBreak

$ReportOutput[$Report]['Summary']

EmailText -LineBreak
EmailText -TextBlock {
"This automation was approved by CAB in $($Type.Ticket). "
"The goal is to keep Group Policies Healthy at all times! "
"In case of issues please contact Przemyslaw Klys "
} -LineBreak
EmailText -Text 'With regards,'
EmailText -Text 'Automated Monitoring'
} -FontSize 10pt
}
}
}
}
```


Tech Savvy Productions

Keep in mind that some of those reports can get really large. For example, the permissions report for 4000 GPOs is about 30MB in size. On the other hand, some other reports are much smaller. This is why there's an option to choose whether to attach a report or not.

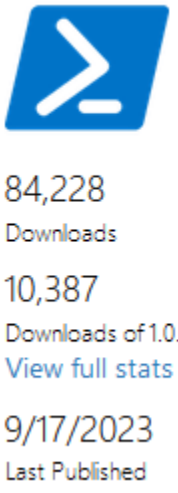
Summary

GPOZaurr is a huge module. It contains a lot of reports, and just a handful of those are shown here. It's almost **20000** lines of code. It can deal with all sorts of **GPO/SYSVOL/NETLOGON** problems you may have. Feel free to explore. On GitHub, the full source code is available (and somewhat readable – one function per file) and about 40 different examples. Not everything may be easy to understand, but I plan to release more blog posts on different ways to deal with issues. What's important to know is that this module will work just fine with just user credentials. Of course, if you've removed authenticated users from a GPO, some reports will skip it, others will mark it as unavailable, but it does work. Of course, fixing issues will require Domain Admin, but that you can do manually – not even running GPOZaurr as Domain Admin.

The code is published on GitHub

Issues should be reported on GitHub

Code is published as a module on PowerShellGallery
https://www.powershellgallery.com/packages/GpoZaurr/1.0.0



The module is signed with a certificate, like any new modules that I create or update.

1. Install-Module GPOZaurr -Force

GO Ahead! Have fun! Make sure to report any issues, or if you feel like something would require covering more ground, let me know. My goal is to have **GPOZaurr** as the only way to deal with **Group Policies**.

Tech Savvy Productions

active directorygpogroup policypowershell

Przemyslaw Klys / About Author

System Architect with over 14 years of experience in the IT field. Skilled, among others, in Active Directory, Microsoft Exchange and Office 365. Profoundly interested in PowerShell. Software geek.

More posts by Przemyslaw Klys

Whoever of you loves life

and desires to see many good days,

keep your tongue from evil

and your lips from telling lies.

Turn from evil and do good;

seek peace and pursue it.

Tech Savvy Productions

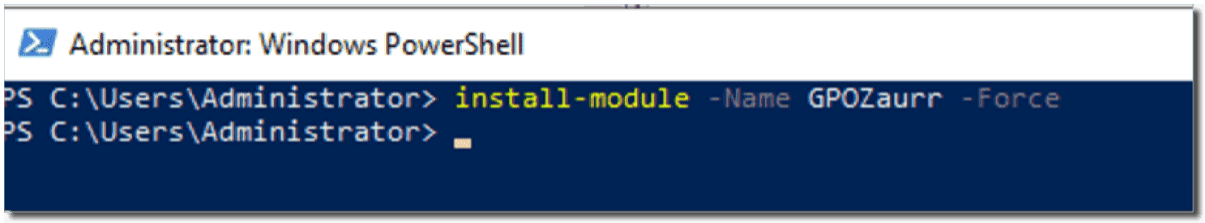
James Rankin

James is a consultant from the UK, specializing mainly in end-user computing, Active Directory and client-side monitoring. When not consulting for james-rankin.com, he can often be found blogging, writing technical articles and speaking at conferences and user groups.

Installing GPOZaurr

You need RSAT installed to support GPOZaurr; after that, you can simply add the module from PowerShell.

Install-Module -Name GPOZaurr -Force

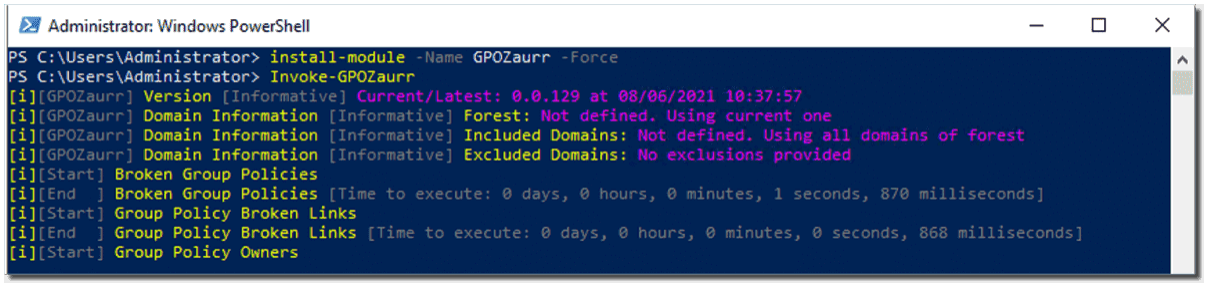


Install GPOZaurr

If you're unable to contact the PowerShell gallery, then you can download the modules on a separate workstation and copy them to the target before installing the module.

Full details are provided on GitHub and in this blog post as well.

The module has many reports that you can use, which are laid out in the second link above. However, in most cases, you can simply generate a full report by running Invoke-GPOZaurr.



Generting a full report with Invoke GPOZaurr

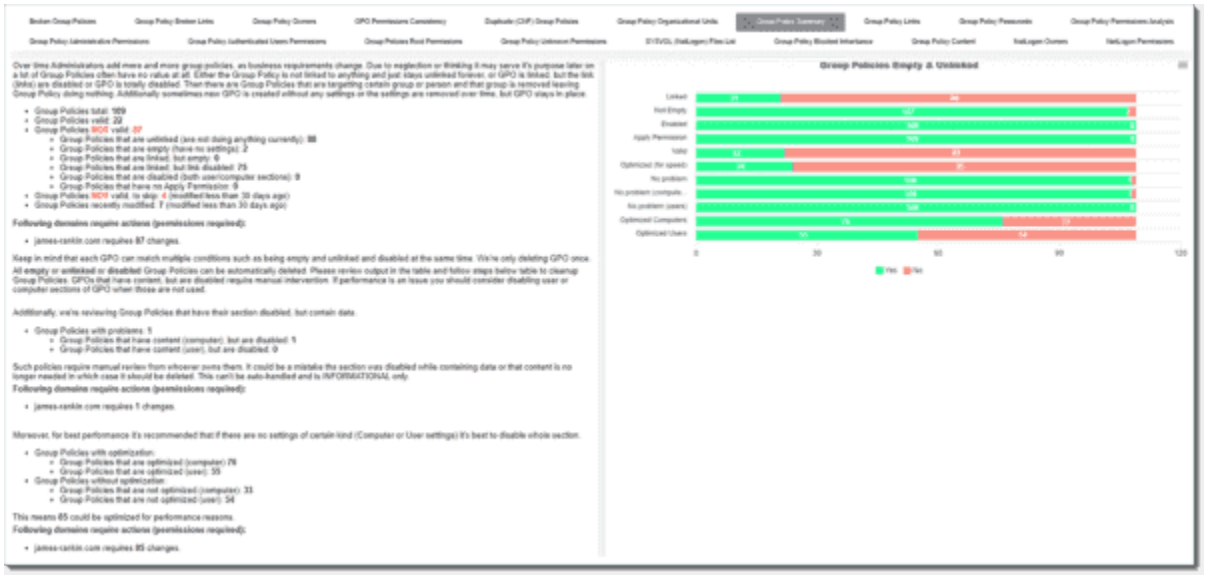
Generating reports

Once the command has finished executing, an interactive HTML report will be generated. This has many output sections that can be browsed, each of which provides different data sets. Each tabbed section is shown below.

Tech Savvy Productions

GPOZaurr report sections

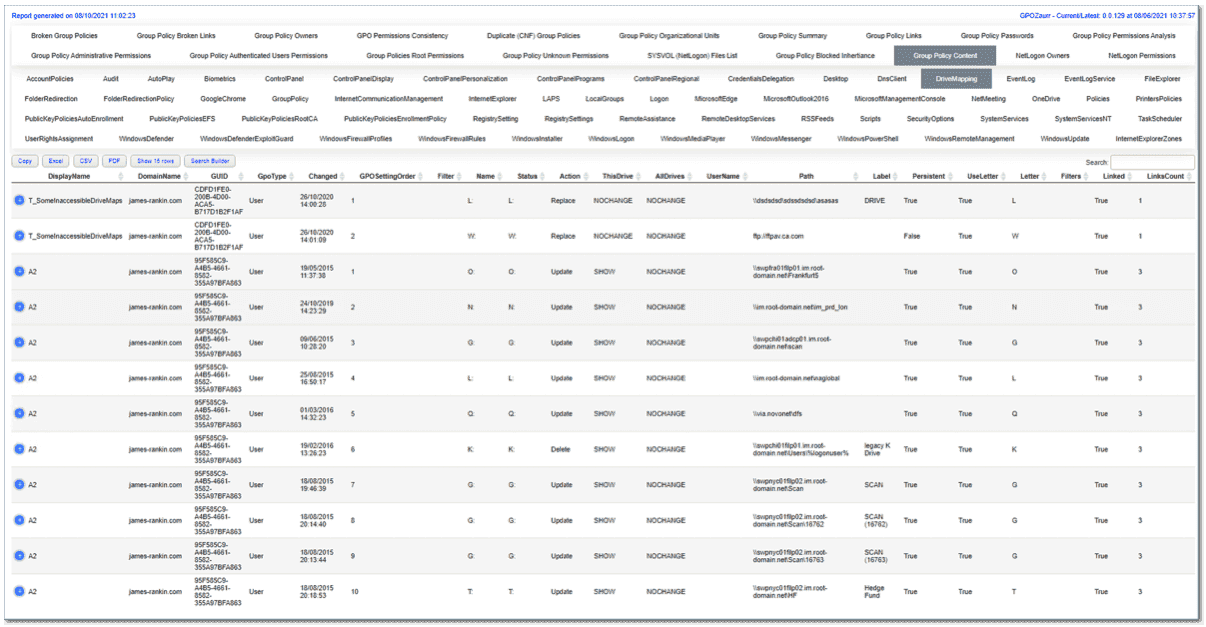
There is also a "Summary" section that provides a handy overview of the report and its recommendations.



GPOZaurr summary report

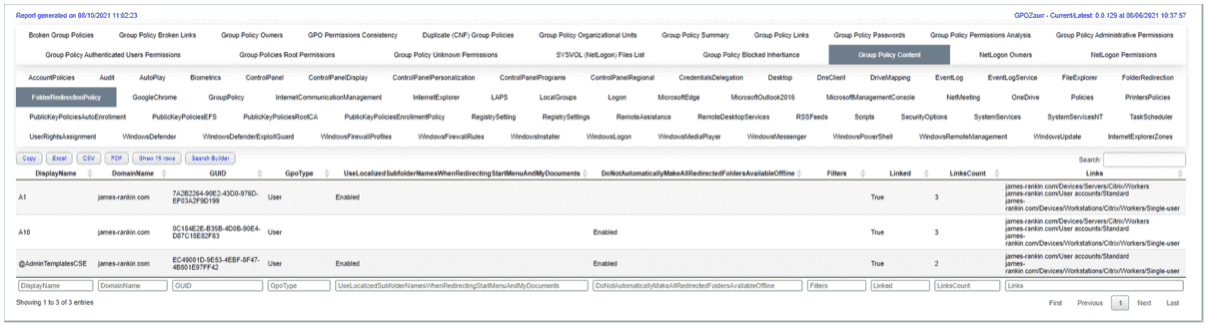
Another excellent feature of GPOZaurr is the ability to export any of the report sections to CSV (for manipulation in Excel), PDF, or HTML format. This allows you to pull out certain subsets of data and then produce more granular information. For instance, I have used it to look at data from the Group Policy Content section and then filter it by Drive Mappings to obtain a list of UNC paths that are being mapped via GP before exporting to CSV. This allowed me to identify any invalid or inaccessible drive mappings being delivered to the users.

Tech Savvy Productions



GPOZaurr report filtered

Looking through the data available in this HTML file gives you fantastic insight into the setup and configuration of your Group Policies. There are some annoyances (such as spelling and grammatical mistakes) that can be chalked up to the author not having English as a first language, but you can easily tidy these up by editing the HTML directly. Of particular interest to me is the Group Policy Content section, which provides a wealth of data that can be perused or exported.



GPOZaurr content view

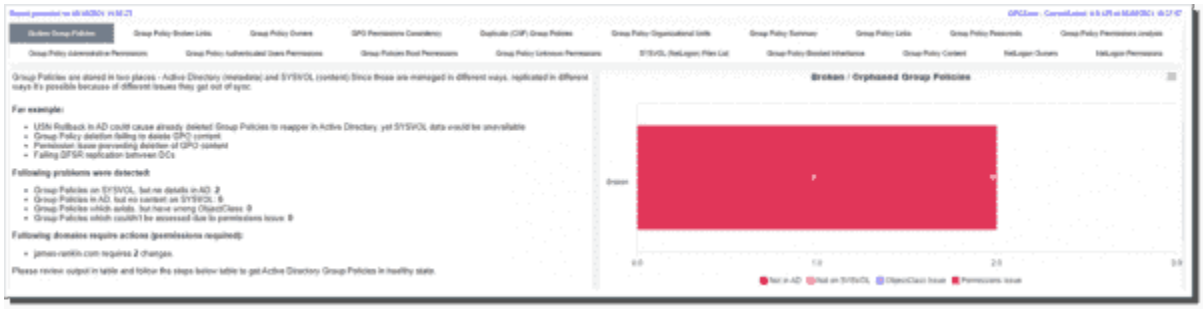
One cautionary mention is that GPOZaurr also provides links to remediation for things such as invalid GPOs or broken links. I would not recommend using this functionality—use the data that has been identified to feed into formal change controls for the removal of GPOs rather than using the tool to do them directly. As such, you should be careful who you provide the raw HTML data to—it is much safer to export the highlights to Excel or PDF for distribution purposes. If you want to avoid the remediation links and distribute HTML, however, you can simply run the tool with the *-HideSteps* parameter specified.

Tech Savvy Productions

Some of the report sections that GPOZaurr outputs are immediately usable, whereas others require more detailed review. It is up to you how you use the data; you can filter and crunch it in any way you please. What I have suggested below is simply a list of guidelines. There are many more things you can achieve!

Finding broken GPOs

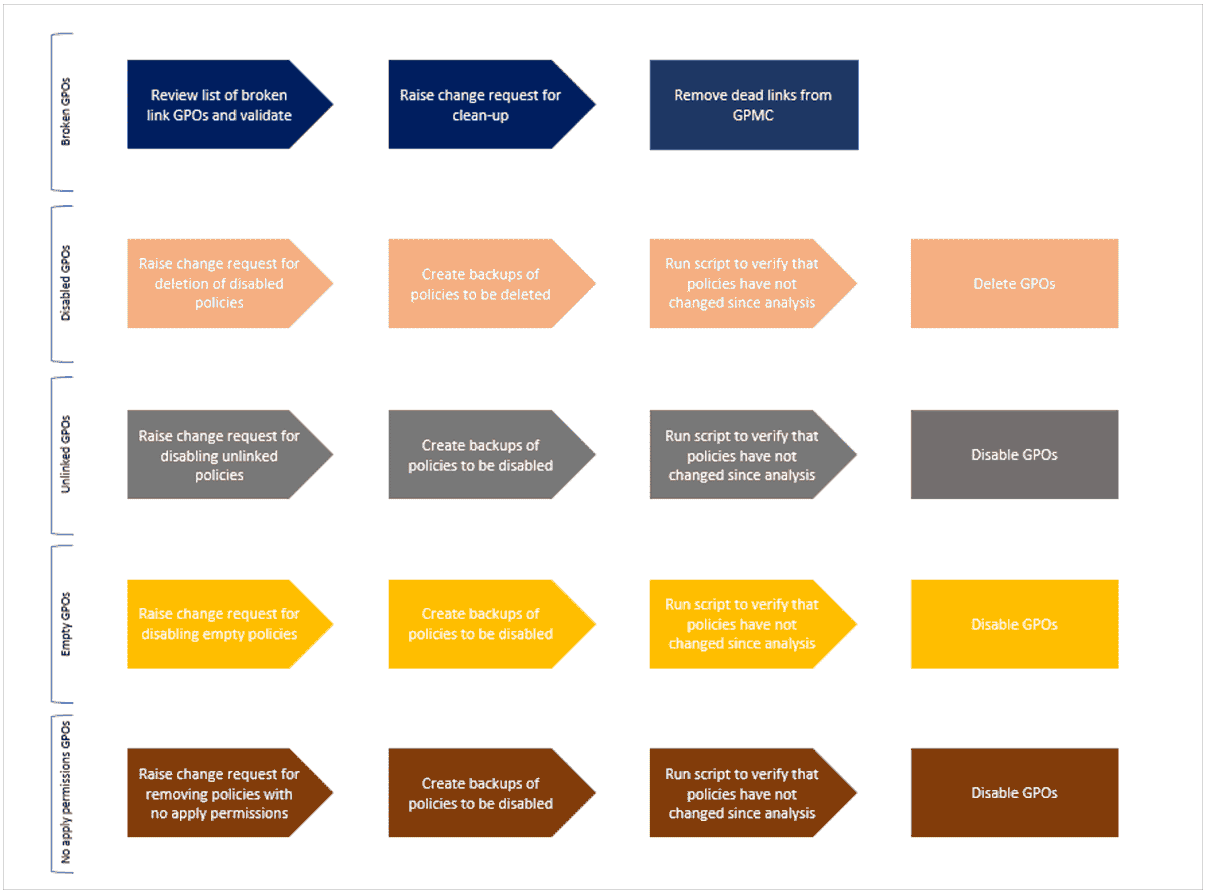
Broken or dead links are instantly identified.



Finding broken GPOs links

The tool also pulls out GPOs that are disabled, duplicated, unlinked, empty, that have sections with disabled content, that users have no permissions to apply, and that have incorrect permissions. This gives a whole load of data that can be quickly verified and fed into a process to remove or disable policies that are not applied for a multitude of reasons. This should be properly validated and change reviewed, as shown in the process below. And it is always prudent to rerun the GPOZaurr utility before making any changes to make sure no one has altered the policies in the meantime.

Tech Savvy Productions

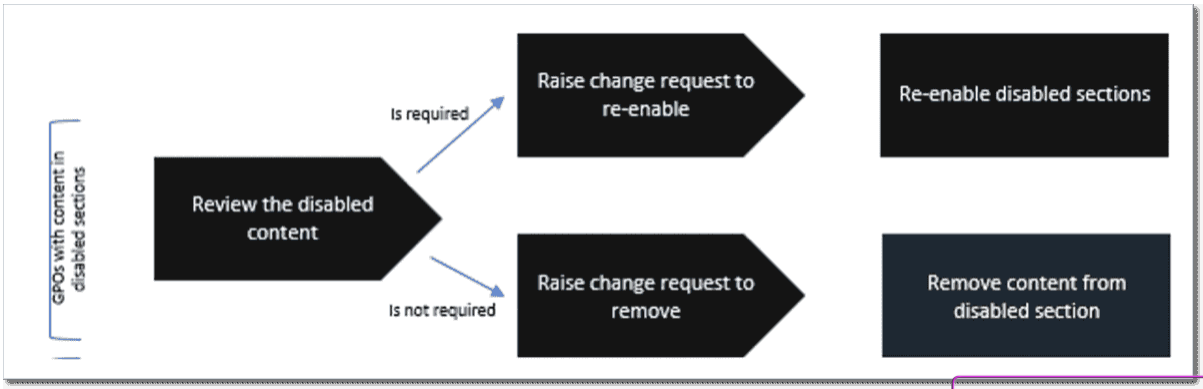


Fixing broken GPOs

In large enterprise environments, this stage of consolidation usually produces at least a few hundred policy objects that can potentially be removed. Once the "quick wins" are done, you can now move on to the slightly more time-consuming phases.

Finding disabled GPOs

First, there will be GPOs identified that have sections (either Computer or User Config) that are disabled but contain content. GPOZaurr will flag these, but they will need a manual review process to verify whether disabled content is required or not.



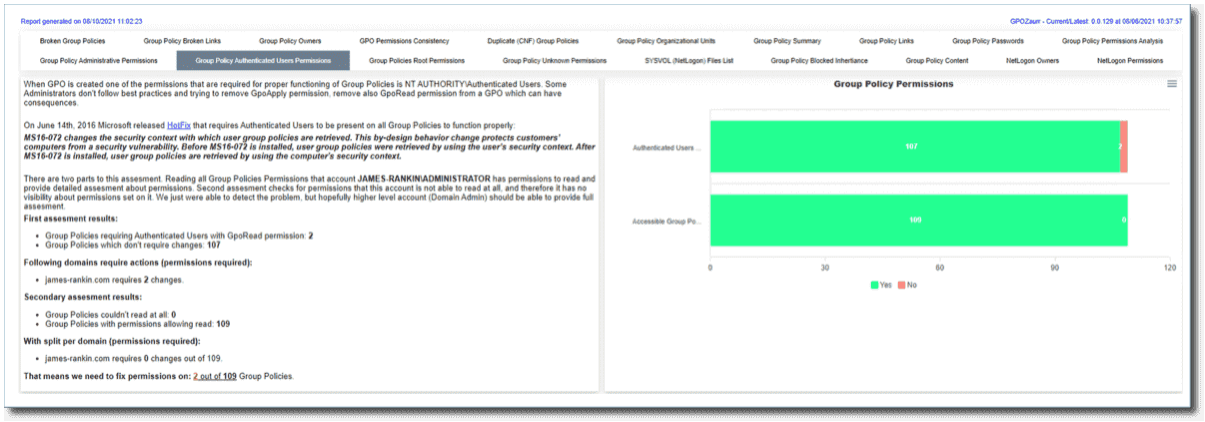
Finding disabled GPOs

Tech Savvy Productions

Finding GPOs with invalid security filters

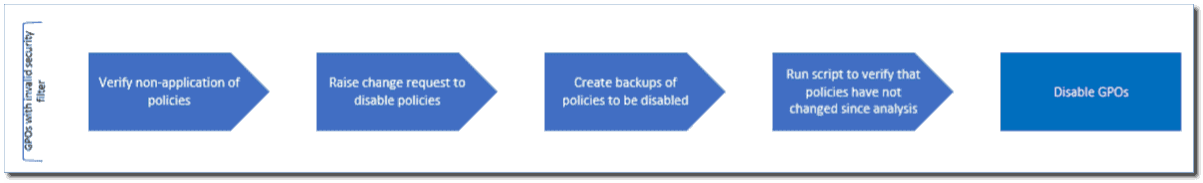
Next is the specter of invalid security filters. Older GP implementations typically have more of these. Several years ago, Microsoft changed the way that the *user* Group Policy is applied via a security update. Some of the *user* processing is now done in the *device* context. This means that if you are filtering a GPO by user or group (and Authenticated Users is not present), then the Domain Computers group (or a list of computer accounts) must be present in order for the filter to successfully apply. GPOs that do not have Authenticated Users or a list of computer accounts specified on the Security Filter cannot be processed.

GPOZaurr outputs a report of "Group Policy Authenticated Users Permissions," which narrows the list down to policies that don't have Authenticated Users present, but each one of these must be checked manually for a group of computers being specified.



GPOs with invalid security filters

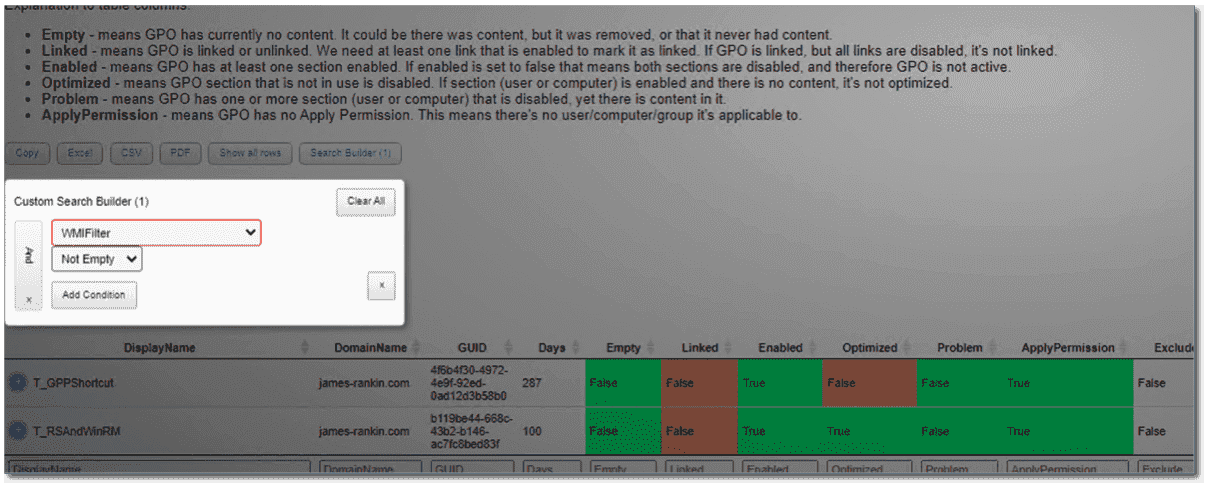
If there are no computer groups or accounts present on the Security Filter, however, this indicates that the GPO has an invalid security filter and should be either remediated or removed.



Fixing GPOs with invalid security filter

You can also apply custom Search Builder conditions to the GPOZaurr data to produce more targeted inquiries. I used this to filter the summary data down into those policies containing WMI filters. It is useful to view and verify these to ensure that they are still applicable and don't take a long time to apply (Product Class is a well-known WMI filter that will cause particular issues).

Tech Savvy Productions



Custom search builder

Finding inapplicable GPOs

For the final part of the medium-term remediation, you can assess whether GPO settings are inapplicable. Many GPOs, for instance, apply only to particular operating systems or software versions (such as settings that only apply to Windows XP and Server 2003). These settings, if present, will still be applied (they are just registry entries), but are essentially ignored. Therefore, it is useful to identify and remediate these.

This is one area where, surprisingly, GPOZaurr does not offer much. However, Microsoft also has the Policy Analyzer tool available, which may help with this in enterprise environments. This tool compares GPOs to look for invalid settings.

Once you have identified any invalid legacy policies (whether by using Policy Analyzer or by manual review), you can then feed the policies into the usual change review process to get them removed.

Tech Savvy Productions

Various tools that allow you to manage and consolidate your Group Policy environment.

In today's article, I will make some general remarks and take a look at two useful GPO tools: Get-GpoReport and Advanced Group Policy Management.

Contents

James Rankin

James is a consultant from the UK, specializing mainly in end-user computing, Active Directory and client-side monitoring. When not consulting for james-rankin.com, he can often be found blogging, writing technical articles and speaking at conferences and user groups.

Many enterprises rely heavily on Group Policy to provide configuration settings to their users and devices. Group Policy, despite being relatively unchanged since 2006, encompasses many configuration items that can be used to push granular settings down to domain-joined devices and/or users. Enterprises often have very complicated Group Policy implementations, which only become more complicated when multiple forests/domains and/or mergers and acquisitions are factored into the equation. In many instances, the complexity of GPO implementation, combined with the fear of inadvertently impacting the user base, leads to Group Policy being left in a sprawling, bloated state that becomes increasingly difficult to manage or unpick.

Tech Savvy Productions

Microsoft's longer-term goal is to move away from Group Policy toward what they call "modern management"—using technology such as InTune and Desired State Configuration rather than the legacy GPO methods to manage their user and device base. For the short term, however, Group Policy is here to stay, at the very least in a hybrid way. As part of a migration away from Group Policy, or just to simplify the day-to-day management and overhead, a consolidation and remediation exercise such as that described in this article is vital.

As well as making management easier and migration more of a realistic possibility, this exercise can also make the processing of policies more efficient, simply by removing unneeded or inapplicable configuration items.

Preparation

If you're in an environment with a sizable number of GPOs (or even if you're not), you may well want to automate as much of this as possible. Trawling through Group Policy Objects manually is a thankless, time-consuming task, so we will suggest automated ways to find the information wherever possible.

With regard to what we'd like to get out of the GPO consolidation exercise, we will attack it from these angles:

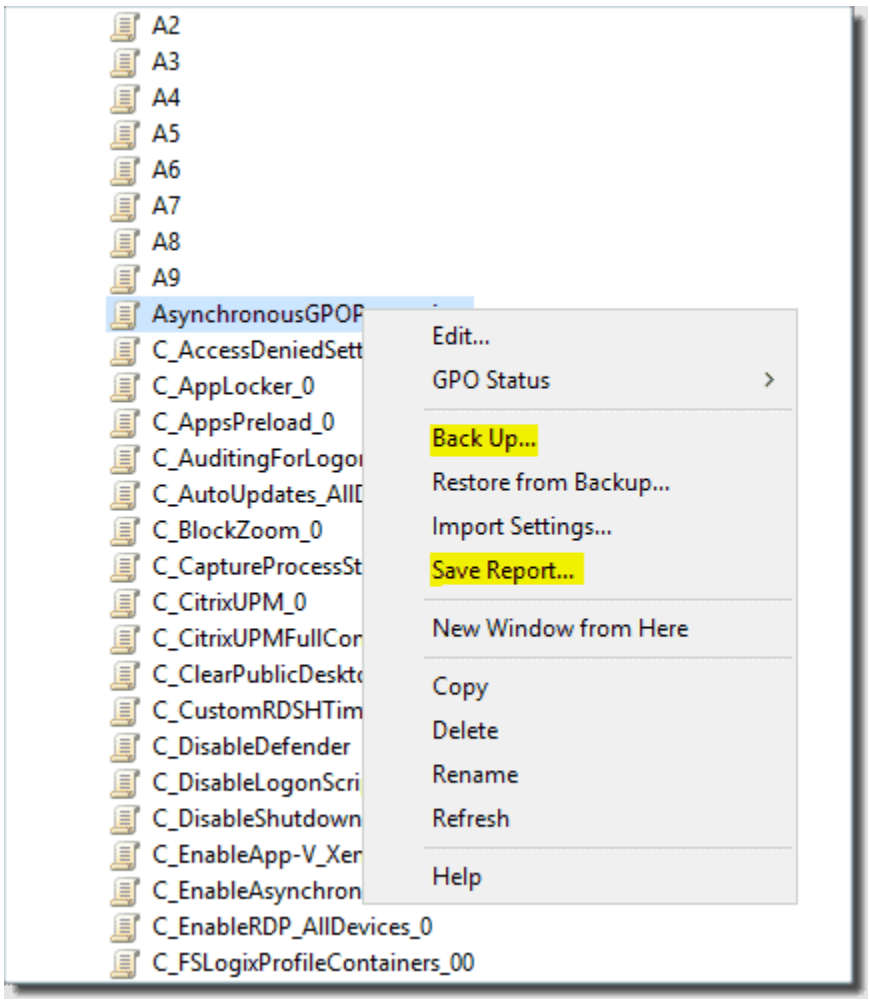
- Remove broken GPOs and dead links
- Remove disabled GPOs
- Remove unlinked GPOs
- Remove empty GPOs
- Identify GPOs with no content
- Identify GPOs with incorrect permissions
- Identify GPOs with inapplicable or legacy settings
- Identify GPOs with invalid security filters

Tech Savvy Productions

Make sure that the user account you are using to do the consolidation exercise has at least Read permissions to all the GPOs in your forest(s) or domain(s).

For the Group Policy PowerShell cmdlets, you need to have access to a machine with the Remote Server Administration Tools (RSAT) installed.

Ensure that you have a backup of your GPOs. Even though we are going through a "read-only" exercise and parsing the data, prudence suggests that you should have a full backup, just in case. You can perform the backup either manually from GPMC by using the "Back Up" or "Save Report" context menu functionality, or use the *Backup-Gpo* cmdlet (which allows all GPOs in a domain to be backed up at once).



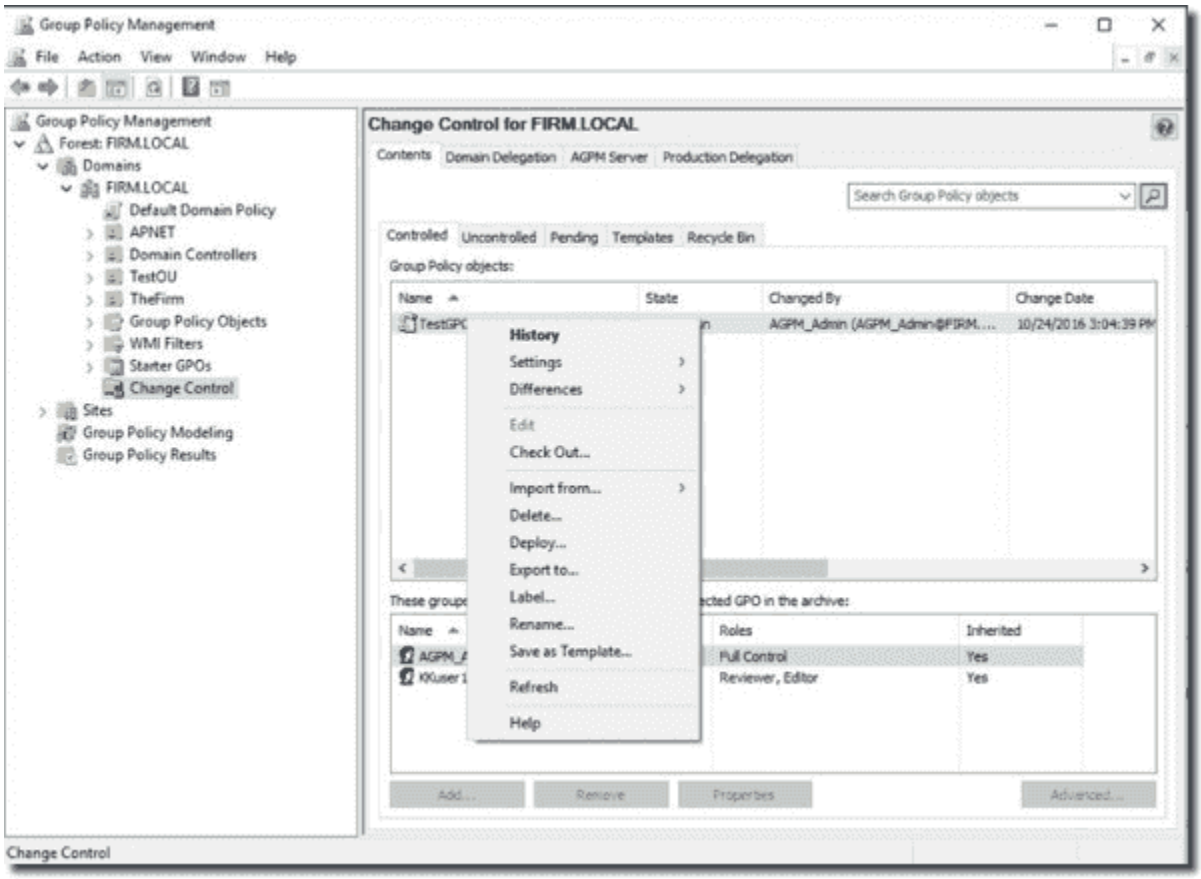
Backing up GPOs

Tech Savvy Productions

There are a couple of tools that can be used for outputting GPO data. The go-to tool is usually the Group Policy PowerShell cmdlets, mainly Get-GpoReport. This can output either HTML or XML reports for all GPOs in a domain. You can combine this with other cmdlets, such as Get-GPPermission and Get-GPO, to produce more targeted data.

Get-GpoReport

One main issue with Get-GpoReport, however, is that it often fails to output an HTML report successfully when run on a large number of policy objects. The XML report works fine; however, this is considerably less readable than the HTML report. Also, even if the HTML report works, parsing this information into actionable data can be time-consuming and may require further scripted manipulation.



AGPM

Tech Savvy Productions

The implementation of AGPMC will provide far more control and failsafes than are currently available within a typical enterprise environment where they use the standard GPMC, as well as allowing more granular reporting and assessment. These changes are crucial to improving the ongoing management of the GP estate.

AGPM simply requires dedicated service accounts and an installation of the console to be implemented.

It is also recommended that the editing of GPOs be locked to AGPMC to prevent users from accessing the policies from other instances of the Group Policy console. If you aren't already using AGPMC, you should start as soon as possible.

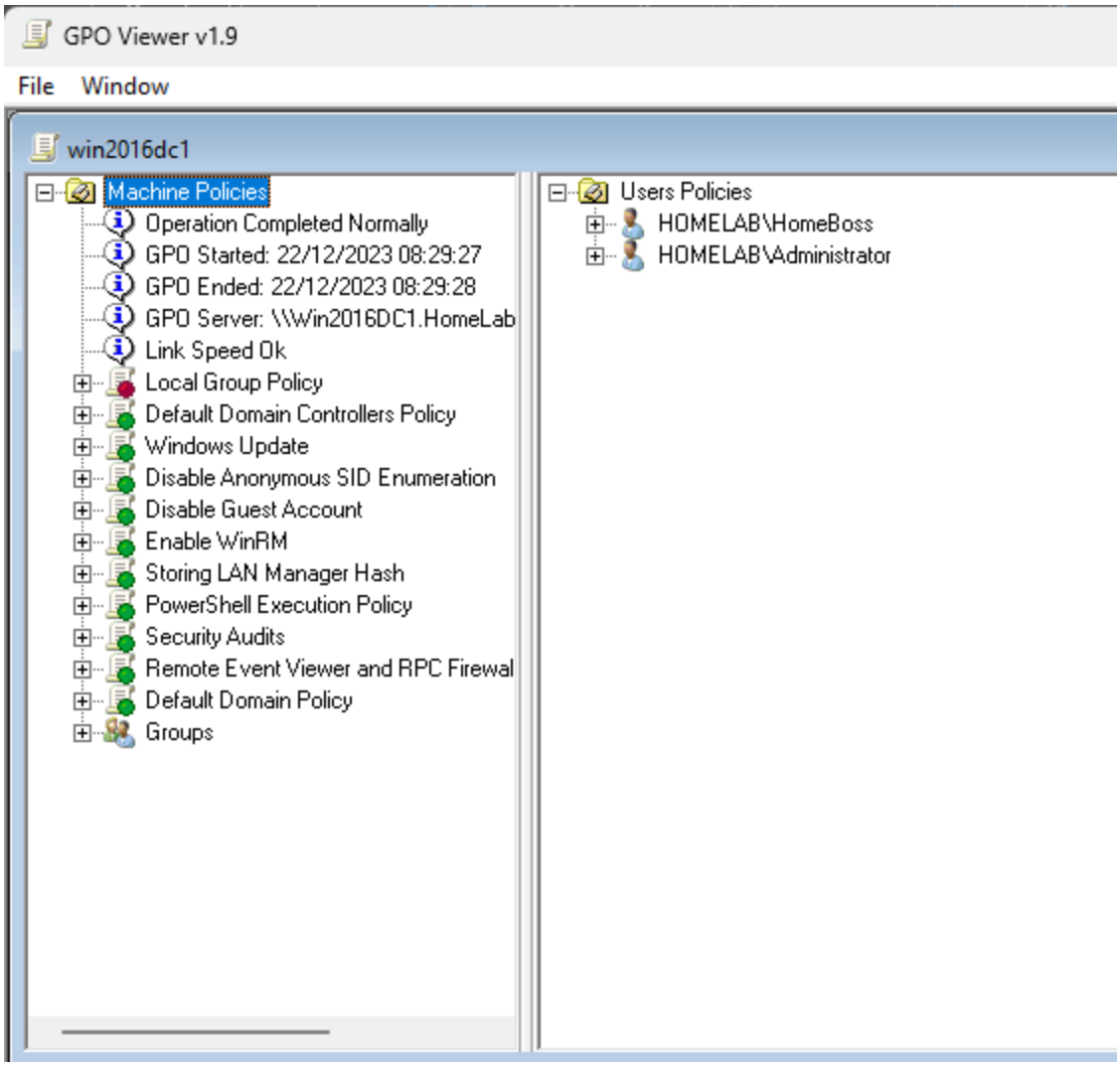
GPO Viewer

Download Now!

This software is Freeware. You use this software at your own risk. I do not warrant this software to be fit for any purpose and I cannot be held accountable for any form of data loss that occurs as a result of using this program, you use it at your own risk.

The functionality of GPO Viewer has now been incorporated in the GPO Explorer option of NetTools

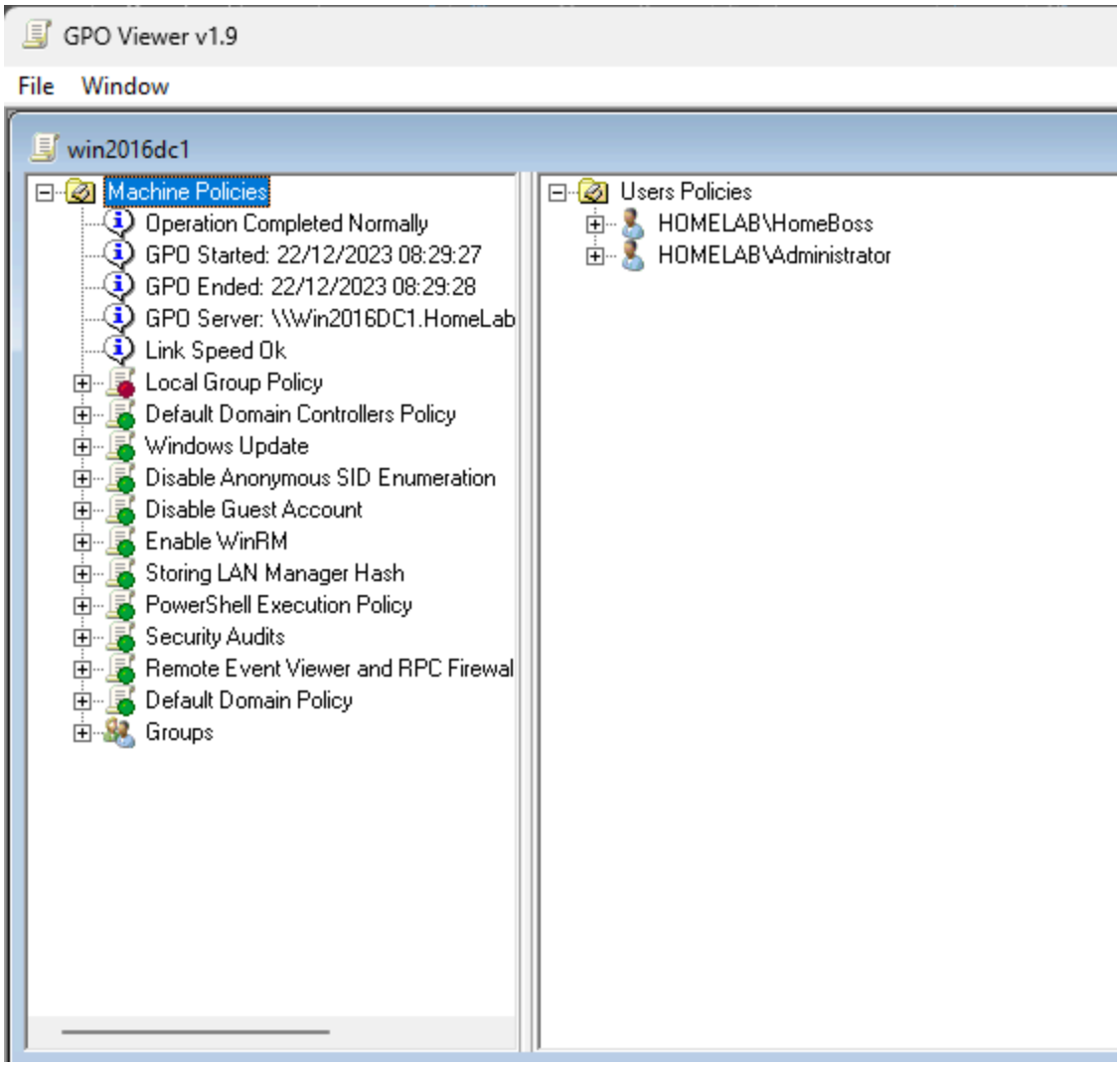
Tech Savvy Productions



Overview

GPO Viewer is a tool I wrote back in 2002 to display which GPOs have been applied to a workstation by reading the results from the GPO engine that are stored in the registry of the machine. It can be used to read the GPO status of both local and remote workstations, the details of each workstation are displayed in a separate window to allow results from different machines to be compared.

Tech Savvy Productions

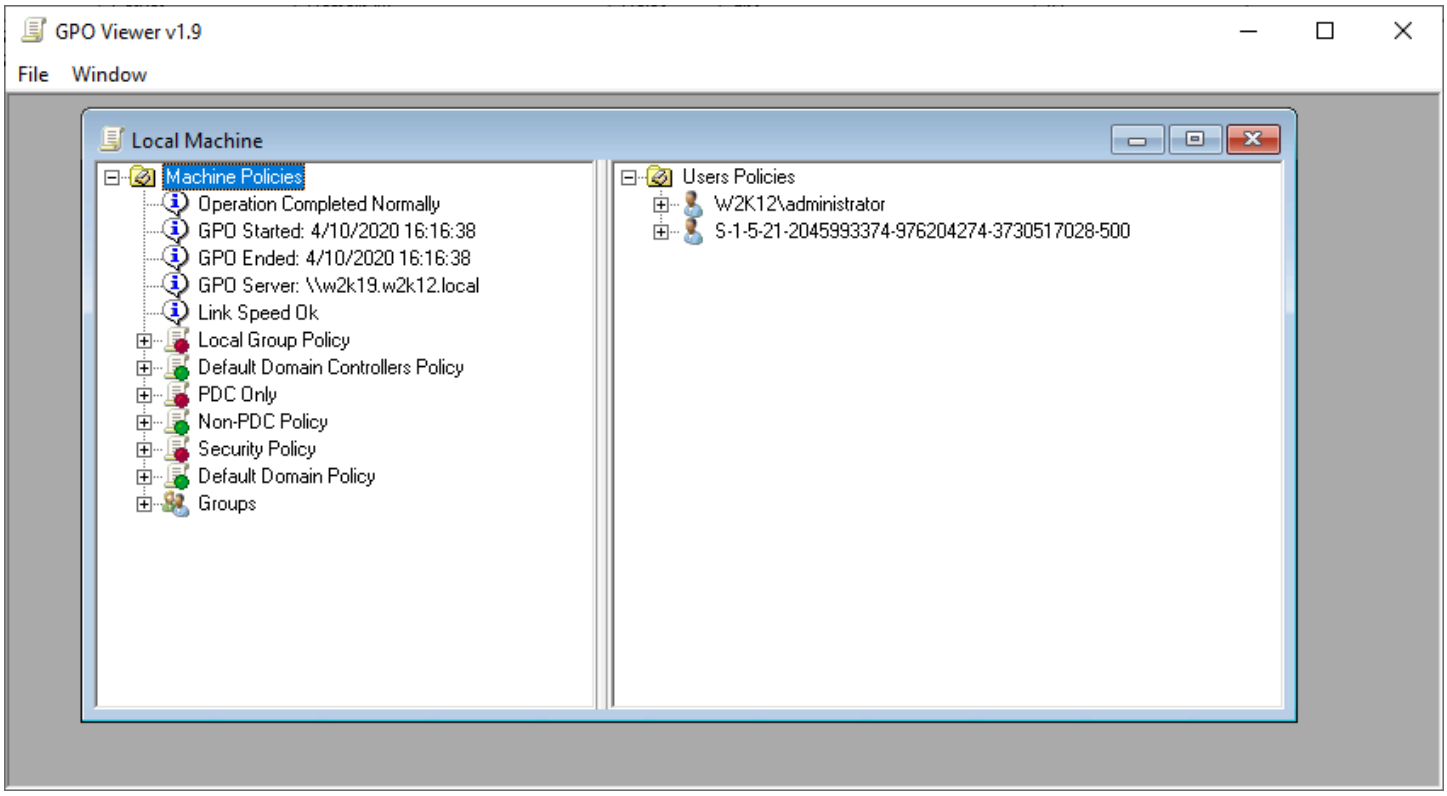


I've dusted it off, updated the graphics, added some additional error checking, and updated some of the terminology to reflect the changes since I wrote it. GPO Viewer provides a graphical user interface to display the policies that have been applied to both the machine and users that have logged onto the workstation. It provides the ability to drill down on each policy and show the details of the GPO engine and in the case where a policy wasn't applied, the reasons why. It also has the ability to save the GPO status details to a file so the details can be reviewed later.

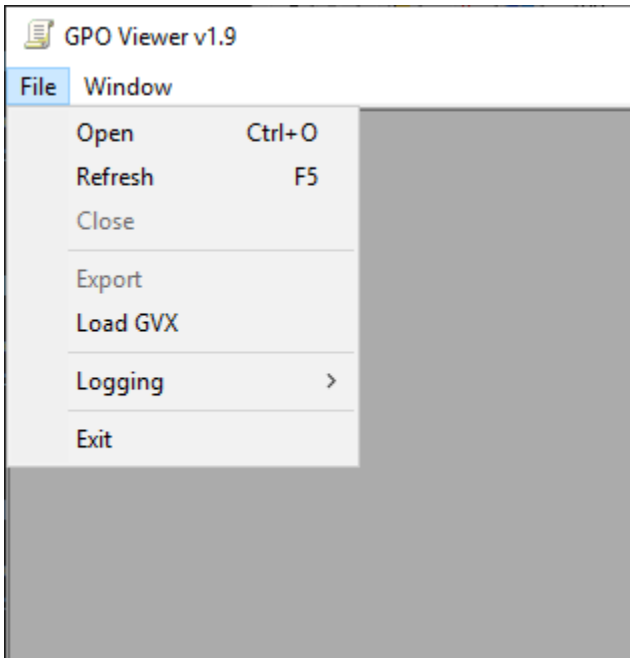
To retrieve the GPO details from a remote machine, the user context running the program must have administrator rights on the remote workstation and the remote registry service must be running.

The screenshoot below shows the results of the a scan against a domin controller.

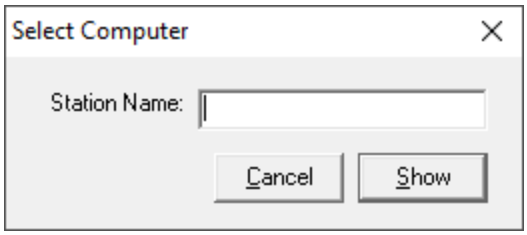
Tech Savvy Productions



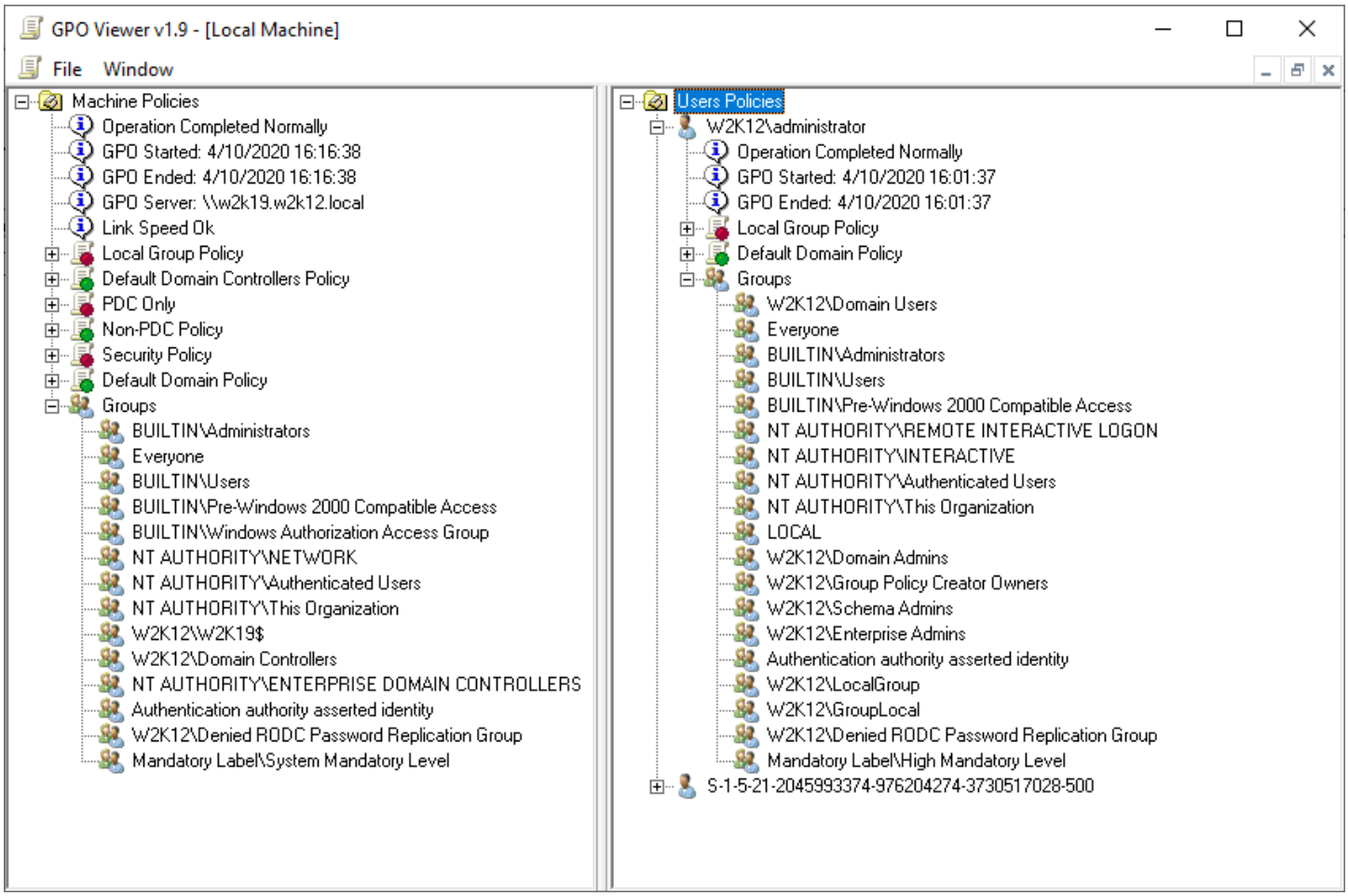
The Open option on File menu is used to select the workstation that use which to scan. In the Select Computer dialog enter the name of the workstation you wish the scan or leave the field blank to scan the local machine.



Tech Savvy Productions



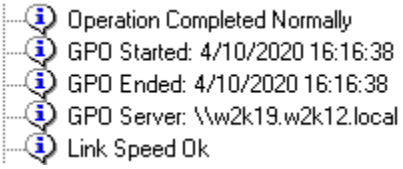
The workstation result window is split into to two sections, the left handside for the machine based policies and right handside for the user policies. Both section provide similar details on the policies that have been applied, while the left handside only display one set of policies that have been applied to the machine, the User section displays all the users that have logged onto the machines and the policies that were applied to that user the last time the policy engine ran.



We will now breakdown each section and provides a little more information of the details that are displayed in each section, the details are the same for the computer and user policies, where there are differences these are called out below.

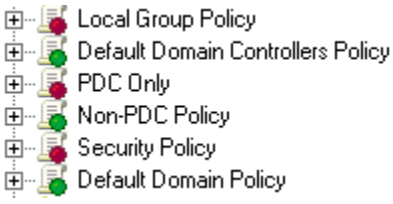
Tech Savvy Productions

GPO Processing Status



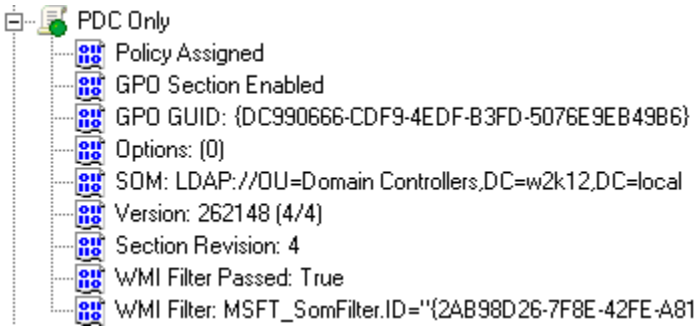
The first three entries provide the details on the status of the policy engine, Operation Completed Normally is displayed if the engine completed without any errors, however, if an error occurred the error details will be displayed. Next is the start and finish times for the last time the Policy Engine ran. The Link Speed option is only displayed in the machine section, and will display a warning if the link speed test has been identified that the machine is connected to the domain controller over a slow link as defined by the policy configuration settings.

Policies Results



The policies that are applied are displayed next, they are listed in the order in which they are applied. For policies that have been applied a green icon is shown next to the name of the policy, if a policy is not applied a red icon is displayed. You can expand the item to display the details of the policy and the reasons why the policy was not applied.

GPO Details



When a policy is expanded it shows the details from the GPO engine. By default the icon colour for these items is blue, however, if an entry indicates that something failed, the icon will be red. The Policy Assigned or Policy Not Assigned shows if the computer or user has been assigned the Appy Policy right. GPO Section shows if the GPO section has been enabled, this is controlled by the GPO Status, i.e. User configuration settings disabled or Computer configuration settings disabled option in GPMC. If a section is enabled but the section doesn't contain any settings then a GPO Section

Tech Savvy Productions

is Empty entry will also be displayed. GPO GUID is the name of the group policy object in AD. Options displays if the policy has enforce option enabled or not, if the policy is set to enforce a padlock is shown next to the status icon. The SOM displays the OU to which the policy is linked in AD. The Version and Section Revision entries display the version details for the policy, which is either the computer or user section version number. WMI Filter Passed show the results of the WMI filter test, if there is no WMI filter applied to the policy the result will be true. The WMI Filter entry display the GUID of the WMI filter that is assigned to the policy.

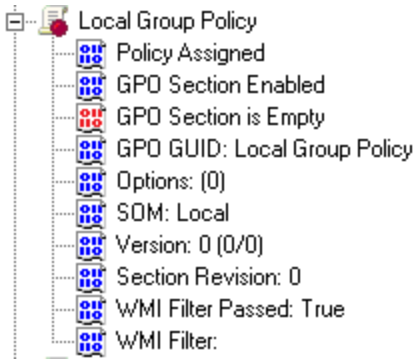
Group Membership

This is the list of groups that the machines, or the user is a member of. This information is based on the machine's or user's access token and not the group membership in AD, as they can be different depending on when the machine was last rebooted, this can effect which policies are applied.

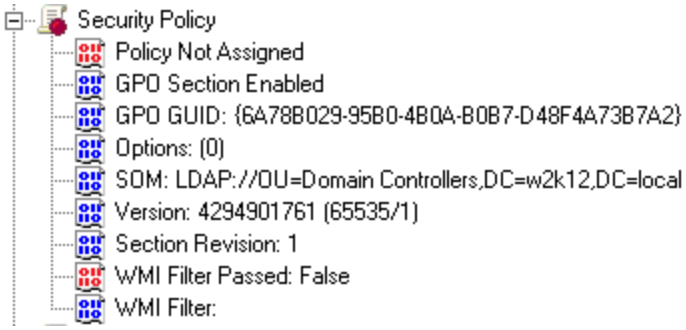
Examples

In this example the section is enabled, however the section is empty, so the GPO is reported as not being applied This common for the local policy.

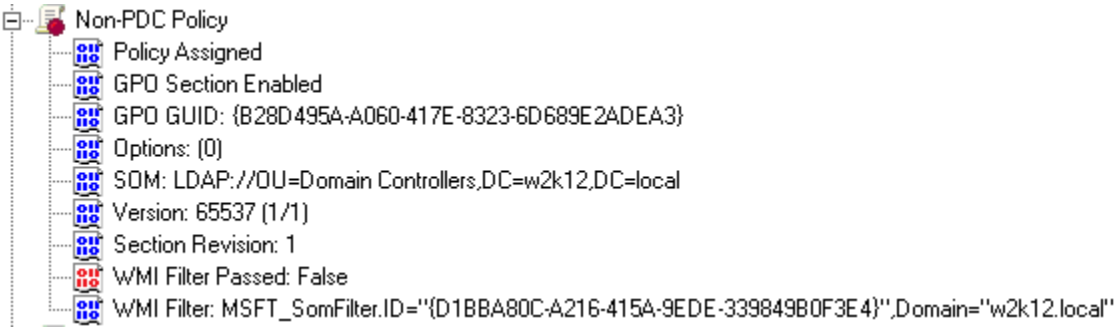
Tech Savvy Productions



In the next example the machine or user have not been assigned the Apply Policy permission and is reported as Policy Not Assigned, as a side effect the WMI filter is also marked as failed.



In this last example the WMI filter test has failed so the policy failed to be applied. The SOM reference for the WMI filter is displayed, you will need to use NetTools, GPO Explorer, WMI Filters, to determine the name and details of the filter.



Export

The File menu has options to export and load the GPO engine results for both the computer and user details to a file, these files have a GVX file extension, these can then be reviewed later or when the machines is not available. As this uses registry fragments to save the details, you still need administrator or SE_RESTORE_NAME and SE_BACKUP_NAME user rights to load and view these files.

Download Now!

Tech Savvy Productions

This software is Freeware. You use this software at your own risk. I do not warrant this software to be fit for any purpose and I cannot be held accountable for any form of data loss that occurs as a result of using this program, you use it at your own risk.

NetTools

The Swiss army knife of AD troubleshooting

https://nettools.net/download/

Features

What's New New

Getting Started

Access Control New

AD Replication

AD Sites

Authentication

Groups

Group Policies

Information

LDAP

Schema

Name Resolution

Network

Tech Savvy Productions

Trusts

Users New

Misc

NetTools contains over 90 different tests and functions, that are grouped into 14 sessions. The high-level details of the individual tests and features is provided below. See NetTools Basics for details on common operations used within NetTools. Details on how to run NetTools can be found here.

Access Control

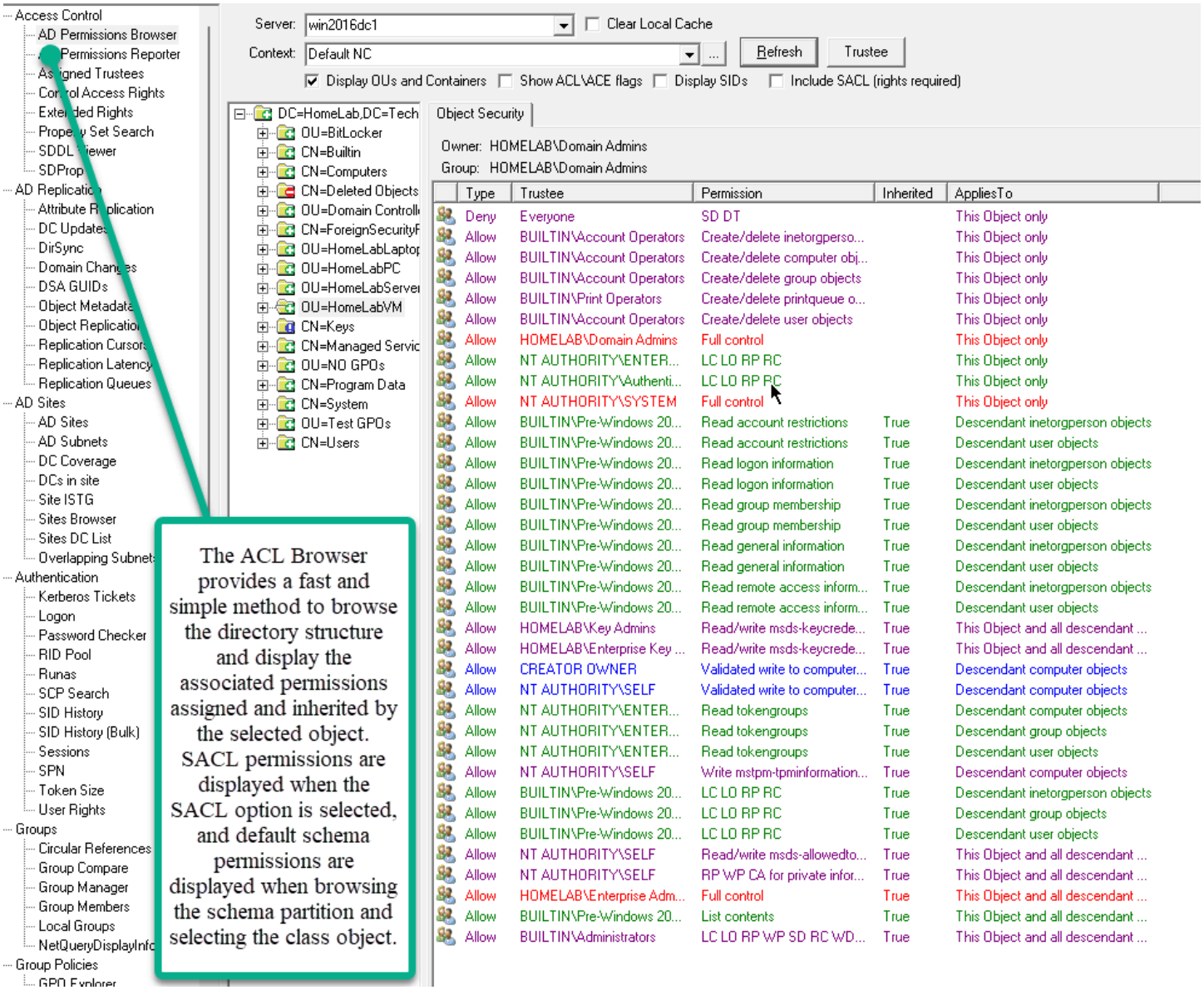
AD Effective Permissions New

A feature in the AD Permissions Browser and Permissions dialog to show what permissions a selected user will receive on the selected objects, this covers the DACL, SACL on all partitions, also the default schema and mailbox permissions. Includes the option to model the impact of permissions changes when developing a delegation model. See AD Effective Permissions

AD Permissions Browser

The ACL Browser provides a fast and simple method to browse the directory structure and display the associated permissions assigned and inherited by the selected object. SACL permissions are displayed when the SACL option is selected, and default schema permissions are displayed when browsing the schema partition and selecting the class object. See AD Permissions Browser

Tech Savvy Productions



AD Permissions Editor New

The AD permissions dialog, provides the ability to edit the permissions of objects in the Active Directory. See How To Edit an AD object's Permissions

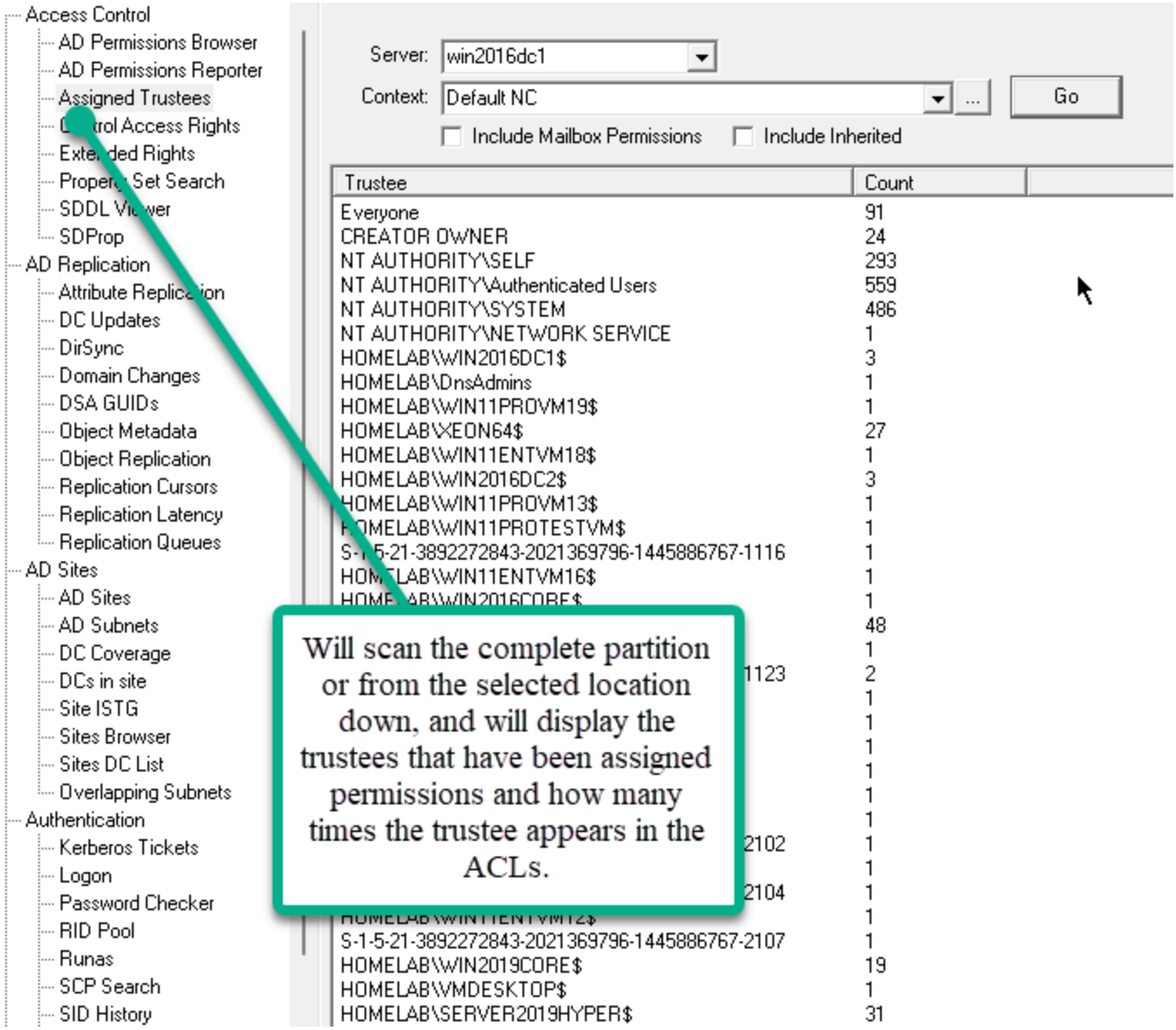
AD Permissions Reporter

A powerful feature to generate reports on who has access and rights in the AD, includes over 30 predefined reports. See AD Permissions Reporter

Assigned Trustees

Will scan the complete partition or from the selected location down, and will display the trustees that have been assigned permissions and how many times the trustee appears in the ACLs.

Tech Savvy Productions



Compare AD Permissions

The capability to compare the permissions of two different AD objects. See Comparing AD Permissions

Control Access Rights

Displays the Control Access Rights that have been defined in the select directory. Selecting a Right will display which attributes the right applies to and the Property Sets the attributes that are included in the Right.

Extended Rights

Display the extended rights that are defined in the directory.

Property Set Search

Will search the property sets for the specified attribute and display all the property sets that include the attribute.

Tech Savvy Productions

SDDL Viewer
A simple option to allow an SDDL string to be displayed in the NetTools permissions dialog See SDDL Viewer

SDProp
This option is used to search for all user and group objects in the domain where the permissions of the object are controlled by the SDProp process. There is also an option to reset the user permissions to restore users that have been orphaned by the process and to allow the SDProp process to reset permissions for users that are still members of a protected group. See SDProp

AD Replication

Attribute Replication
Is used to confirm that the specified attributes of the selected object have been replicated across all domain controllers.

DC Updates
Will display the number of updates that have been processed across all the domain controllers, in the forest or domain controllers hosting the selected domain context. See DC Update

DirSync
Utilizes the LDAP DIRSYNC server-side control to display the updates that have been made on the domain controller. See DirSync

Domain Changes
Displays the objects that have been updated on the domain controller based on the USNChanged attribute. See Domain Changes

DSA GUIDs
Will display all the DSA GUIDs and Invocation GUIDs registered against the selected server.

Object Metadata
Will display the directory services meta data for the selected object. See Object Metadata

Object Replication
Used to confirm that objects and attributes are replicating across the selected domain controllers. See Object Replication

Replication Cursors
Will display the directory replication cursors for the selected domain controllers.

Tech Savvy Productions

Replication Latency
This option provides the ability to test the time taken to replicate a new object across all the domain controllers in the select partition. The test will create the selected object type and then delete the object once the test is complete

Replication Queues
This option displays the directory replication queues on the specified domain controller. Domain Admins or Replicating Directory Changes right is required to display the contents of the replication queues.

AD Sites

AD Sites
A simple DNS test to find which domain controllers will be used based on a machine's IP address or AD site name. The returned domain controllers will be tested to confirm that they respond to ping, LDAP,, and GC ports.

AD Subnets
Will display which AD site the specified IP address to assign to, or if you paste a list of IP addresses into the main pane, the AD site for the corresponding IP addresses will be displayed. See AD Subnets

DC Coverage
A simple DNS test to return what AD sites are serviced by the specified domain controller.

DCs in site
another simple DNS test to return what domain controllers are registered against the specified AD site. The Site Name can be selected from a dropdown list.

Site ISTG
It will display which domain controllers is performing the ISTG role for each site. See Site ISTG

Sites Browser
A combined view lets you view the AD Site details in a simple hierarchical browser. The following details can be displayed, AD Sites, Subnets, Site Links, Domain Controllers, Query Policy, connections, Downstream Partners, Naming Contexts, Licensing, Site coverage, Link Costs, NTDS settings, and test domain controller connectivity. See Site Browser

Sites DC List
Displays the list of domain controllers in the specified forest, for each domain controller the site name, default domain context, roles, FQDN, and IP address is displayed. See Sites DC List

Overlapping Subnets
This will scan the IP addresses defined in the forest and display any IP address ranges that overlap another IP address range. See Overlapping Subnets

Authentication

Kerberos Tickets
Provides the ability to display the Kerberos tickets that are associated with the current user context, or a specified Session. It is possible to purge individual or all tickets, and request a new ticket based on the specified SPN. See Kerberos Tickets

Logon
A simple test using the LogonUser API and allows you to specify the API parameters to test different authentication methods and types. i.e. GPO User Rights configuration. If the login is successful, the corresponding groups and privileges will be displayed. See Logon

Password Checker
A crude password checker to check if the specified password is being used by a list of accounts. The list is added by pasting the list of samaccountname into the pane.

RID Pool
Used to display the RID pool allocation and the next RID for all the domain controllers in the forest. The current RID pool master and the next RID pool allocation is also displayed. See RID Pool

Runas
A simple test using the CreateProcessWithLogonW API to execute a program using the defined set of credentials. This was one of the first options added and could do with some love to update the form.

SCP Search
An option to allow you to search the directory of the specified SCP. with the ability to search based on the service name or the GUID of the service.

Sessions
Will display the existing logon sessions that exist on the local machines and display the processes that are associated with the logon session. See Sessions

SID History
An option to display and manage the SID history against a single user or group object.

SID History (Bulk)
A bulk update option to allow the SID History to be set on a number of objects based on a semi-colon-separated input file. The option uses the DsAddSidHistory API which has a number of prerequisites which are tested by the validation step before you can import and update the SID History of the specified objects. See SID History Bulk

SPN
An option to search the directory for the specified Service Principal Name, the search uses the sPNMappings settings to search for alternative service names against the host. See SPN

Token Size
This option will display the token sizes for all objects that match the specified search criteria. Once the list is returned it is possible to explore which direct and nested groups contribute to the overall token size. See Token Size

User Rights
This will display the groups and privileges that are assigned to the user context in which NetTools is running. See User Rights

Groups

Circular References
Used to find any circular references or infinite loops in group membership See Circular References

Group Compare
Provides the ability to compare the group membership between two users. There are a number of different name resolution and comparison options available. See Group Compare

Group Manager
An option to allow the membership of a group to be updated, allows changes to be specified as SamAccountName, SID, UPN, email, or DN input. The changes are pasted into the right-hand pane. See Group Manager

Group Members
An option to display the members of a group, including recursive across nested groups, displaying which group delivered the membership.

Tech Savvy Productions

Local Groups (NetGroupEnum)
An option to display the members of the local groups associated with the specified server. See Local Groups

NetQueryDisplayInfo
Will display the local or global groups associated with the specified server using the NetQueryDisplayInfo API.

Group Policies

GPO Explorer
Provides similar functionality to GPMC to browse the GPO defined in the specified forest, and also includes the test functionality of GPOTool.exe. Provides the ability to view GPO allocations, settings, permissions, view the contents of the registry.pol file, and test the GPOs providing similar functionality as GPOTool. See GPO Explorer

Information

Server Info
Based on the NetServerGetInfo API, this option provides the ability to display the configuration information of the selected server

User Info
Based on the NetUserGetInfo API, this option provides the ability to display the details of the selected local user account.

LDAP

Compare Objects
Provides the ability to compare differences between two objects or the changes that have been made to a single object. See Compare Objects

LDAP Browser
This option allows you to browse the contents of a directory in a three-pane view. Including the ability to restore deleted AD objects. See LDAP Browser

Tech Savvy Productions

LDAP Performance
This option performs a number of LDAP directory read operations and displays the time taken to perform these operations. The number of time the tests are run can be configured and Min, Max, and Avg is displayed. See LDAP Performance

LDAP Ping
This option uses raw WinSocket packet injection to simulate the CLDAP protocol and allows the NeutralizeNT options to be bypassed. Still, there isn't much call for this option now that NT4\Windows 2000 hybrid domains have pretty much disappeared! See LDAP Ping

LDAP Search
A powerful and feature-rich LDAP client providing user-selectable data type decodes, server-side control, LDAP session options control, LDAP browser, display filters, save favourites, filter string substitution for common data types, table view, queries based on multiple inputs, LDAP filter wizard, batch multiple queries and feed the result into subsequent queries, create write\update queries, and much more. See LDAP Search

Manage Lists
A sub-function of the LDAP Search feature, which allows lists of data to be set up and then used by the display filters. See Display Filters

Object Count
An option to count the number of different types of objects that exist under the selected OU structure. Selectable object types for Users, Groups, Computers, Active Users and all objects. See Object Count

Schema

Schema Class Browser
Displays the schema classes as defined in the selected LDAP directory. Provide a list of the defined schema classes, when selected it shows the attributes that are included in that class, as well the source class of the attributes. It also displays the hierarchy for the selected class. See Schema Class Browser

Schema History
This option displays the updates that have been performed on the schema and the name of the corresponding update based on the internal database and user-defined entries in the NetTools.ini, i.e. Windows 2008, 2012, 2019, Exchange CU update, and third-party schema providers etc. See Schema History

Tech Savvy Productions

Schema Versions
This option displays the current version of various schema, features and functions, included, Forest, Domain, and Domain Controller Functional Level, RODC, Schema Version, Exchange Schema, Forest, and Domain level, attribute and class counts against each Domain Controller in the forest. Ideal for confirming that a schema update has been completed and replicated across the forest. See Schema Version

Name Resolution

DC Resolution
This option provides the ability to check the consistency of the DNS, DSAPI, LDAP configuration for the domain controllers in the forest. There is also the option to complete a port scan to confirm if the ports are available. The list of servers and ports to be tested can be user defined. The server list is defined by pasting the list of server IP addresses to be tested. See DC Resolution

DsGetDcName
This option provides the ability to call the DsGetDcName API directly with user-specified parameters. The DsGetDcName is part of the NetLogon service and used is to find domain controllers in the forest\domain. See DsGetDcName

NetGetDcName
The option allows the legacy NetGetDcName API to be called with user-specified parameters. See NetGetDcName

Local Groups
The option uses the legacy Windows networking NetServerEnum API to display the groups on the local or remote servers in the domain.

WINS Lookup
A command line style function that let you query WINS servers. Supports user-defined record types in queries.

Network

Certificate Checker
This feature allows you to verify the certificates that are used to protect websites and show the results of the revocation checks. See Certificate Checker

Tech Savvy Productions

HTTP Headers
An option to display the HTTP headers that are returned by the website, with the option to follow or not follow directions See HTTP Headers

IP Geo Location
An option to query the ip-api.com API service to query the GEO location of a specific IP address or name. See IP GEO Location

Ping
A multiple-threaded ping function that allows you to ping multiple IP addresses at once. The devices that are to be ping are pasted into the pane, the list can be IP addresses, FQDN, or shortnames. See Ping

Trace Route
A multiple-threaded trace route function that checks all hops simultaneously to provide the fastest possible results. See Trace Route

WhoIs
An option to query the WHOIS database for the details of the specified domain name, with an option to follow referral to sub WHOIS database authority. See WhoIs

UNC Check
This option will test the specified UNC path and confirm each component of the path is correct including, name resolution, ping, share existence, permissions and directory is searchable. See UNC Check

URL Check
Combines the HTTP header, IP GEO Location, Trace Route, WhoIs, Ping and DNS resolution tests against the specified domain name, the referral and redirects are defined by the individual tests. See URL Check

Trusts

Domain Tree
This option will display the list of the domains and domain controllers in the specified forest.

DsTrust
This option will display the trusts that are returned by the DsTrust API against the specified server.

Tech Savvy Productions

LsaTrust
This option will display the trusts that are returned by the LsaOpenPolicy and LsaEnumerateTrustedDomains APIs against the specified server. Administrator rights are required for this API.

Users

Group Changes
An audit function to display the group membership changes that have been performed on the selected user. This will display which groups the user has been added and removed from. See Group Changes

Last Logon
This option will display the last logon details for the specified user against all the domain controllers containing the user, including the last logon time per DC, last password change, lock time, and bad password time. There is a single-click button to unlock the account. There is also an option to trace back through the event logs on the domain controllers and the member servers in the authentication request to find the details of why an account has been locked out. This functionality requires Security Log read rights and is dependent on the event log details not being lost by event wrapping. See Troubleshooting account lockouts

Last Logon Time
The option will query all the domain controllers in the domain to get the LastLogon attribute and display the latest time. The option support querying multiple users, the list of users is pasted into the pane. See Last Logon Time

Locked Accounts
The option will display all accounts that are currently locked in the specified domain and provides the option to bulk unlock selected accounts. See Locked Accounts

NetUserEnum
This option will allow the browsing of the local groups on member servers and the users assigned to the groups.

Org Structure
An option to browse the organisational structure of the specified user based on the user's manager and direct report attributes, for the selected user a common set of attributes is displayed, and if defined the associated thumbnail picture is also displayed.

Tech Savvy Productions

Reports New

This option will show all the direct and indirect reports for the selected user. See Reports

Search

This option uses ANR based searches to search for the specified user or other objects in the domain or forest. From the search results it is possible to link the select user or object to other options using the context menu. See User Search

Top Quota Usage

The top quota assignments are displayed against the selected domain context or all contexts in the domain. It's also possible to search for the quotas of a specifying user.

User's Groups

This option will display the user's group membership as returned by the TokenGroup attributes associated with the user.

User's Membership

This option will display the nested group membership of users, and which nested groups contributed to the user's group membership. See User's Membership

Misc

ASN.1 Viewer

This option is used to display ASN.1 data structures, support for DER, PEM, PKCS#7, and PKCS#12 file formats, and manual input in hex and base64 formats. Includes support for common x.509 field types. See ASN.1 Viewer

Base64

An option to convert text, GUID, or Hex to Base64 and back. There is also an option to create a new GUID if required from the context menu. See Base64

Clipboard Formats

This is a simple option to display the details of the data that is currently held in the clipboard buffer.

Error Messages

An option to display the error messages associated with an error number based on the DisplayMessage API. There is also an option to display LSA and LDAP based errors.

Tech Savvy Productions

GUID Search
Provides the ability to search for a GUID against a number of GUID stored in the directory. See GUID Search.

Mail Conflicts
A rather specific option to test for potential mail address conflicts that may occur during a domain migration.

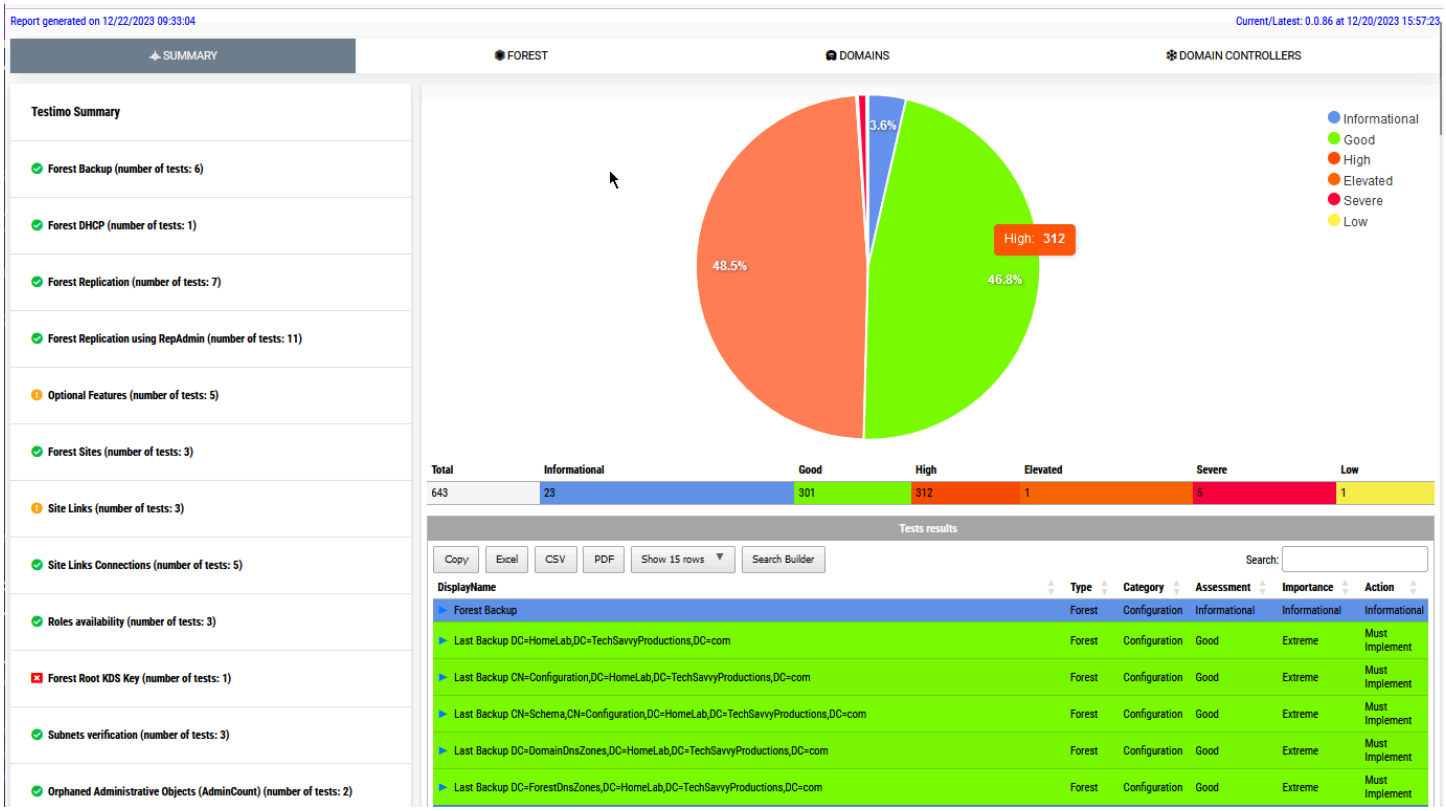
Mail Unique
A rather specific option to test for potential mail address conflicts that may occur during a domain migration.

Relative Identifiers
This option search against the selected domain for the specified RID.

SID Converter
An option to resolve a name to the corresponding SID and visa versa, the number of different formats are displayed. See SID Converter

Time Converter
An option to convert time and date into a number of different formats. Supported formats include Generalized Time, Int64, Azure format and returned across local and UTC time zones. See Time Converter

https://evotec.xyz/what-do-we-say-to-health-checking-active-directory/



What do we say to health checking Active Directory?

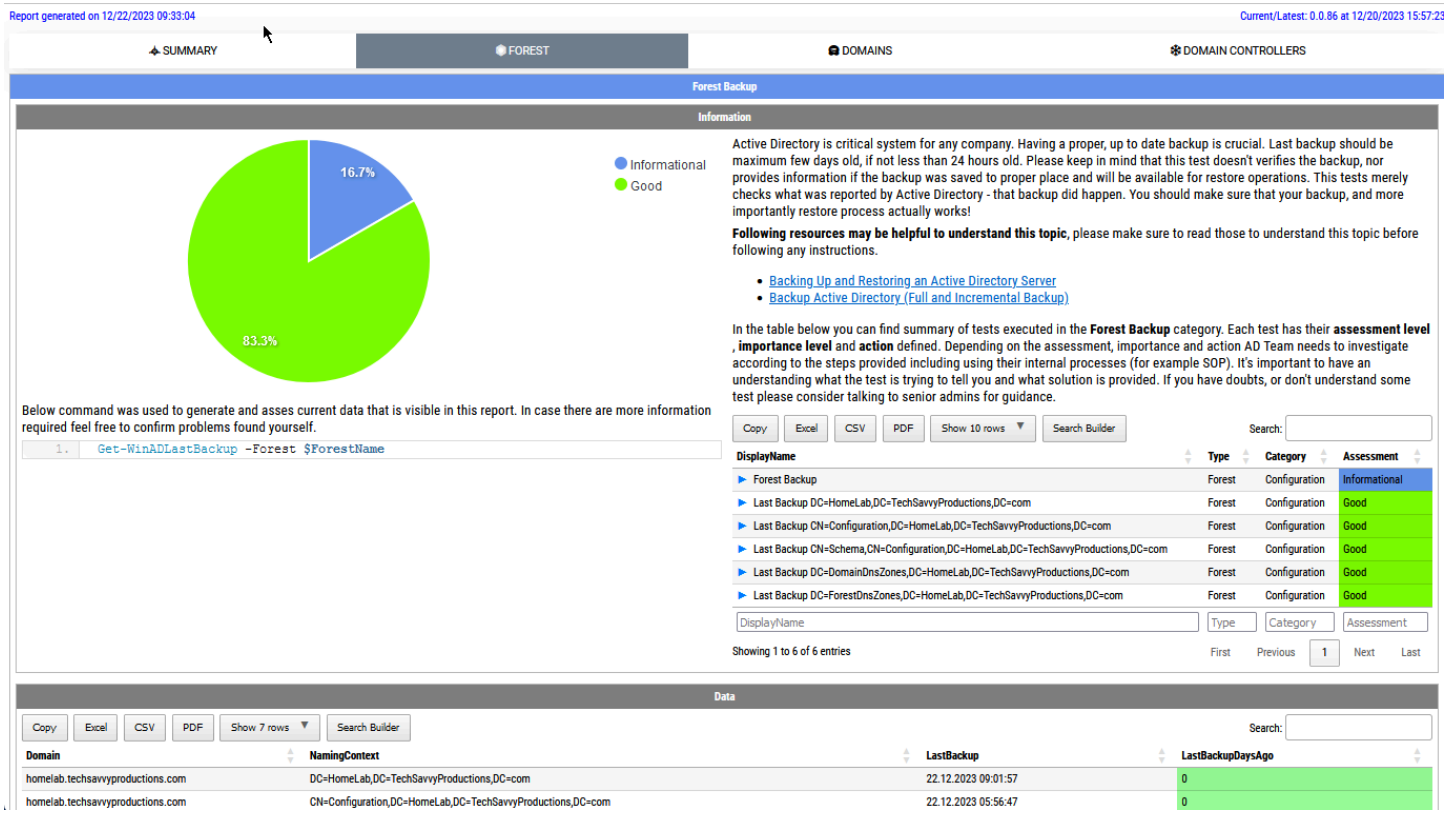
Setting up a new **Active Directory** is an easy task. You download and install Windows Server, install required roles and in 4 hours or less have a basic **Active Directory** setup. In an ideal world that would be all and your only task would be to manage users, computers, and groups occasionally creating some Group Policies. Unfortunately, things with **Active Directory** aren't as easy as I've pictured it. **Active Directory** is a whole ecosystem and works well ranging from small companies with ten users to 500k users or more (haven't seen one myself – but so they say!). When you scale **Active Directory** adding more servers, more domains things tend to get complicated, and while things on top may look like they work correctly, in practice, they may not. That's why, as an Administrator, you need to manage Active Directory in terms of its **Health and Security**. Seems easy right? Not quite. While you may think you have done everything, checked everything, there's always something missing. Unless you have instructions for everything and can guarantee that things stay the same way as you left them forever, it's a bit more complicated. That's why **Microsoft** delivers you tools to the troubleshoot your Active Directory, such as **dcdiag**, **repadmin** and some others. They also sell

Tech Savvy Productions

monitoring solutions such as **Microsoft SCOM** which can help and detect when some things happen in your AD while you were gone. Surely there are some **3rd party** companies give you some tools that can help with a lot of that as well. Finally, there is lot of folks within the community creating PowerShell scripts or functions that help with some **Health Checks of your Active Directory**.

Active Directory Health

For years I've been copying/pasting some stuff from different blogs trying to find what I've misconfigured or what I am missing. Alternatively, I used now discontinued **Microsoft Active Directory** Replication **Status Tool** that was excellent looking way telling me what doesn't work when it comes to **Active Directory Replication**. Do you remember it?



Some months ago, I saw on Twitter how **Bill Kindle** wrote **Pester** based **Active Directory** test. If you don't know **Pester**, it's a **test framework** for **PowerShell scripts**. He used it in the non-conventional way of testing infrastructure with it. It was a bit eye-opener for me even thou I know some people have done similar things before. At that time, I took his code, done some improvements, few small adjustments, and released to **PowerShell Gallery** under **PesterInfrastructureTests** name. This gave me the ability to use it freely without the need to copy/paste any code. I was using it for confirming that my or my Clients **Active Directory** works correctly. It covered some basic tests, but it was handy. It made me forget about **the AD Replication Status Tool**, and I could quickly check if everything is running correctly after

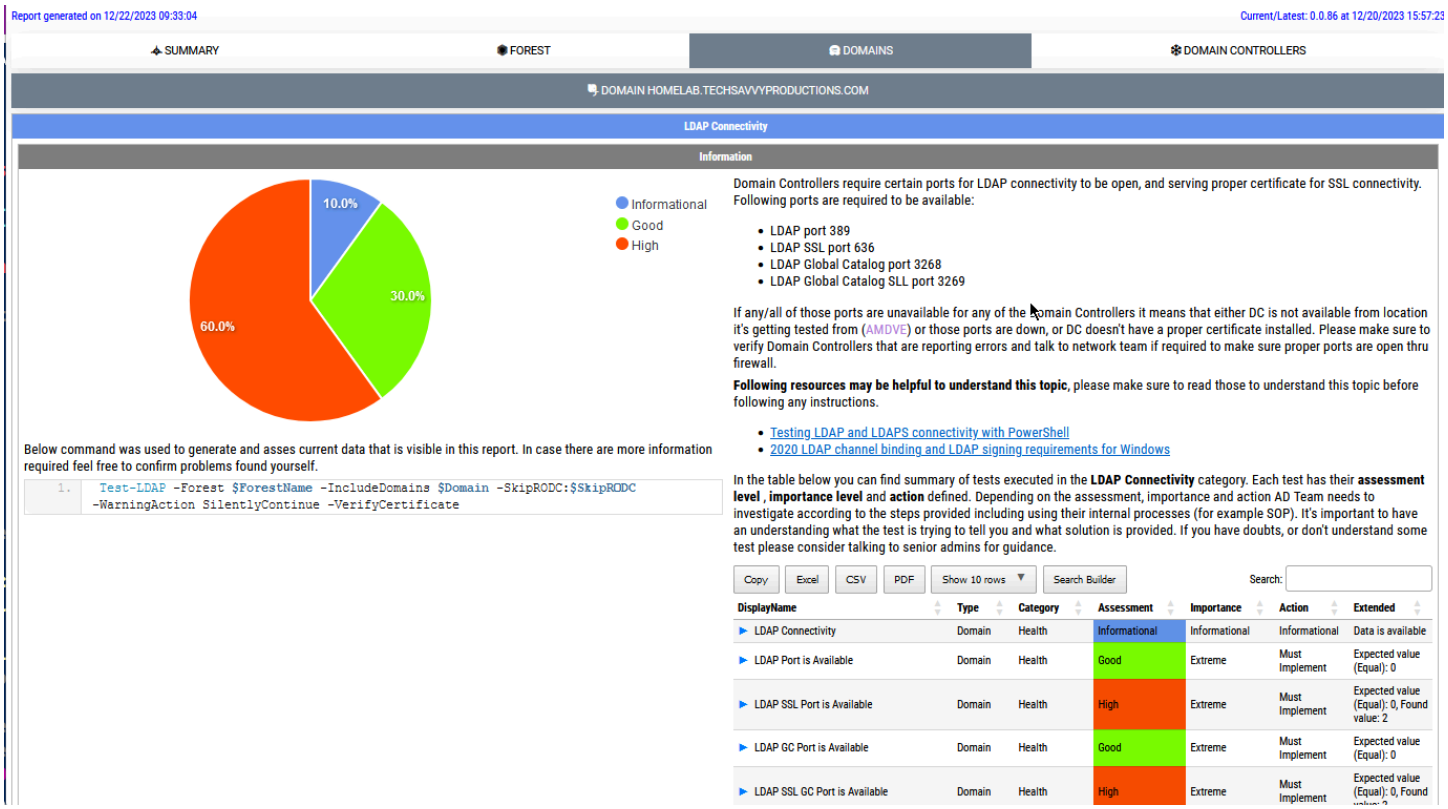
Windows Updates. Let's see how **PesterInfrastructureTests** looks like and what it does, shall we? All we have to do is install it using **Install-Module** which downloads and install **PesterInfrastructureTests**. After that, we run a single command that targets **Active Directory Forest** scanning all Domains and all Domain Controllers doing some basics tests.

1. # First Install Module if you don't have one
2. # Install-Module PesterInfrastructureTests -Force -AllowClobber
3. Test-ADPester

What I love about it, is that with one command **Test-ADPester** it delivered me the status of my **Active Directory**. Not a lot of it, but enough to tell whether my **Active Directory** is healthy or not, with virtually zero effort on my side.

See what I mean? So much red, some yellow warnings and some white ones. Oh well, when I see red, that means it's terrible, and I should jump in to fix things. I could probably do some try/catch, prettify it a bit and enjoy its functionality for more years to come. Unfortunately for me, a few months ago on a trip to **PSConfEU (PowerShell Conference Europe)**, I've spent 8-hour driving with one of my colleagues **Mateusz Czerniawski** talking about **Health Checks for Active Directory**, and how cool would it be to have something similar to what **DBAChecks** offers to **Microsoft SQL Community**. Cool idea, right? He then followed thru by releasing **Active Directory Health Check List** on **Github** – a curated list of health checks that someone can take and build upon. It was just a list, but something one can rely on to develop their stuff.

Tech Savvy Productions



Useful AD Health and Security Materials

Me, being me, and with my tendency to reinvent the wheel, I thought that I need something more then **Pester** checks and that my **PesterInfrastructureTests** is not going to cut it. I've decided to do a lot of research trying to find what others have done in the area and while there are lots of materials and approaches all require manual steps to start, or check an only small part of **Active Directory**. Following is a list of some things I found useful and things that gave me ideas to build something of my own.

- Active Directory Checklist by Mateusz Czerniawski
- pChecksAD by Mateusz Czerniawski
- PSADHealth by Mike Kanakos and Stephen Valdinger
- ADPosh and his other repositories by Chad Cox
- An Active Directory Health Check PowerShell script by Jeff Wouters
- Active Directory Health Check, Audit and Remediation Scripts by JEREMY SAUNDERS
- Microsoft Active Directory Health Check PowerShell Script Version 2.0 by Carl Webster / Jeff Wouters and Michael B Smith
- ACTIVE DIRECTORY HEALTH CHECK & OFFICE 365 by JUSTANGUY

- Active Directory Health Check
- VBS Script that check's your Active Directory Health
- Active Directory Health Check by Sukhija Vikas
- How to Drive Revenue with Active Directory Health-Checks by Thomas Mitchell
- Active Directory Operations Test by Irwin Strachan
- Test-ActiveDirectory by Mark Wragg

And that's just a list of resources that cover at least 1 test. I've gone thru plenty of blog posts, websites, scripts, **GitHub** trying to asses what people think **Health Check** means for **Active Directory**. **Arnaud Loos** list is by far my most favorite, and the most comprehensive list I found. It's a gold mine!

- AD Health & Security Check-up by Arnaud Loos

Testimo - Basic features

Of making many books there is no end, and much study wearies the body.

Now all has been heard;

here is the conclusion of the matter:

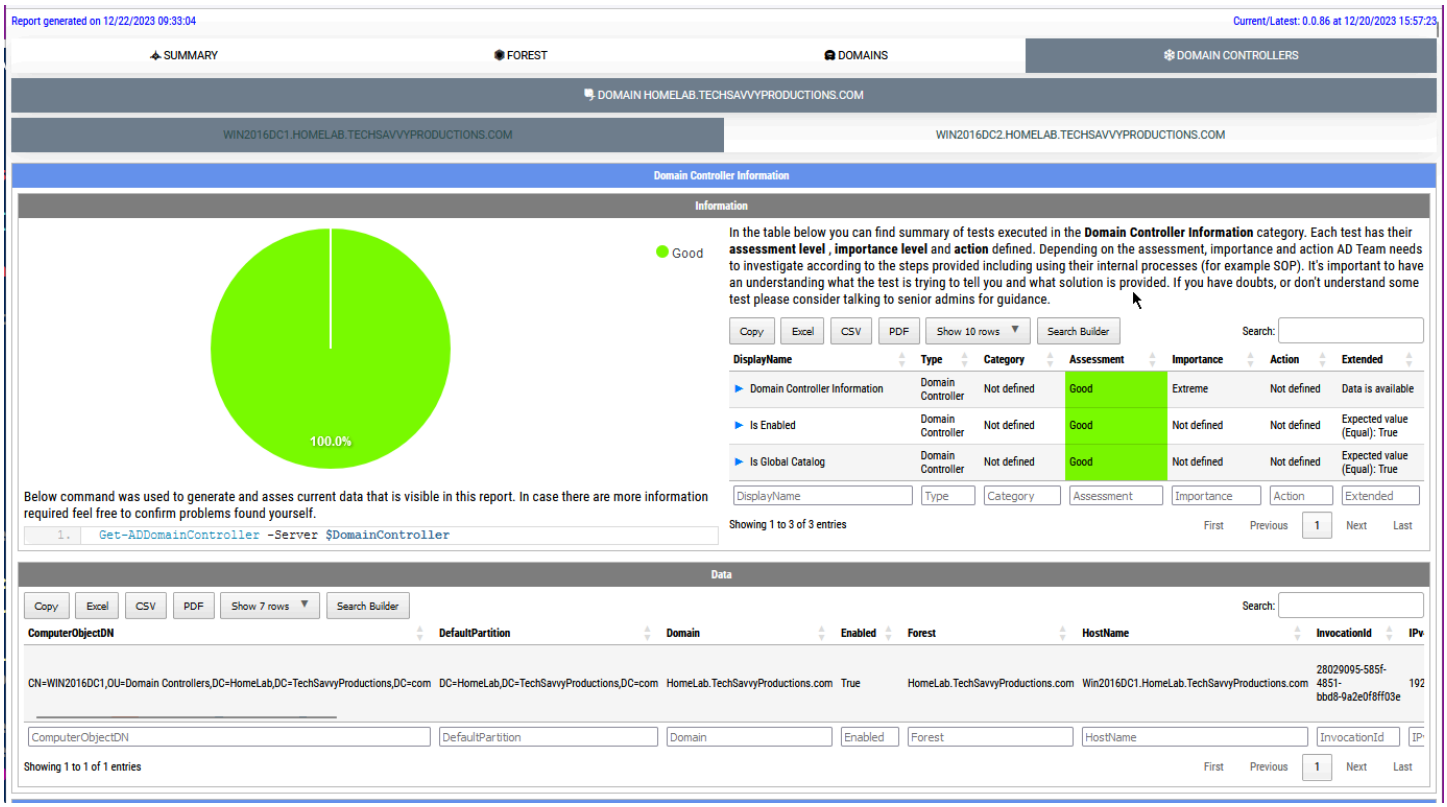
Fear God and keep his commandments,

for this is the duty of all mankind.

For God will bring every deed into judgment,

including every hidden thing,

whether it is good or evil.



Now all I have to do is reinvent the wheel and build something of my own. But what should it be? How should it work? Well, after a few weeks of working on it, changing how things are done, how things are working I've decided that it should at least cover these goals:

- It should allow for testing Forest
- It should allow for testing all domains within a forest
- It should allow for testing all domain controllers in all domains
- It should be easily extensible in adding more tests
- It should allow changing settings on a global way (disable/enable test)
- It should allow changing each test separately (think password policy differences between different companies)
- It should allow excluding Domains and Domain Controllers from tests
- It should allow easy installation
- It should allow easy use, intuitive use out of the box
- It should have tests based on settings rather then a snapshot/baseline comparison only
- It should deliver Pass/Fail per test
- It should allow for reporting with extended information and email status if required

Don't get me wrong, I did have those plans as above, but I've changed my mind probably 50 times while working on a solution. Today is just a day I am comfortable enough to share it with you and **ask for feedback**.

Testimo is **PowerShell Module** that I wrote to help me asses **Active Directory Forests** that I manage frequently or in-frequently. I often consult for **MSP** companies that manage multiple Clients, each with different **Active Directory**. Alternatively, I get ad-hoc requests for help, and so I need something that can help me quickly verify where the problem may be, without knowing how someone built their environment. It **needs to be flexible** enough to tell me where a possible problem may be without spending much time trying to find that problem. Usually, when I get such request you need to go thru lots of different areas by hand, and since I'm pretty lazy my happiness level goes down, and we don't want that right? That's why I've created Testimo, and I hope you will enjoy it as much as I do. It's by no means module that covers everything, but I hope it will be able to help me do my job better and faster. Let's see if it can help you? But before I jump into it, I wanted to add that **Testimo** is joining my two other **AD-related** modules that you may find useful.

- What do we say about Active Directory Documentation
- The only PowerShell Command you will ever need to find out what did what in Active Directory

Keep in mind that while the blog posts cover some functionality, I do work on those modules from time to time adding new features. It's best to track **GitHub** if you want to know what changes or what's added. I don't do blogs for every release.

- **PSWinReportingV2** – home for Find-Events and general reporting on Events (not only related to AD)
- **PSWinDocumentation** – home for documentation of AD / Office 365 – engine only
- **PSWinDocumentation.AD** – home for a dataset for Active Directory
- **PSWinDocumentation.O365** – home for a dataset for O365
- **PSWinDocumentation.O365HealthService** – home for an Office 365 Health Service

So what **Testimo** can do? I've defined set of different tests with a very short overview what is the expectation from a test

- **Forest** Backup – **Verify** last backup time should be less than X days
- **Forest** Replication – **Verify** each DC in replication site can reach other replication members

Tech Savvy Productions

- **Forest** Optional Features – **Verify** Optional Feature Recycle Bin should be Enabled
- **Forest** Optional Features- **Verify** Optional Feature Privileged Access Management Feature should be Enabled
- **Forest** Optional Features – **Verify** Optional Feature Laps should be enabled Configured
- **Forest** Sites Verification **Verify** each site has at least one subnet configured
- **Forest** Sites Verification **Verify** each site has at least one domain controller configured
- **Forest** Site Links – **Verify** each site link is automatic
- **Forest** Site Links – **Verify** each site link uses notifications
- **Forest** Site Links- **Verify** each site link does not use notifications
- **Forest** Roles **Verify** each FSMO holder is reachable
- **Forest** Orphaned/Empty Admins – **Verify** there are no Orphaned Admins (users/groups/computers)
- **Forest** Tombstone Lifetime – **Verify** Tombstone lifetime is greater or equal 180 days
- **Domain** Roles **Verify** each FSMO holder is reachable
- **Domain** Password Complexity Requirements – **Verify** Password Complexity Policy should be Enabled
- **Domain** Password Complexity Requirements – **Verify** Password Length should be greater than X
- **Domain** Password Complexity Requirements – **Verify** Password Threshold should be greater than X
- **Domain** Password Complexity Requirements – **Verify** Password Lockout Duration should be greater than X minutes
- **Domain** Password Complexity Requirements – **Verify** Password Lockout Observation Window should be greater than X minutes
- **Domain** Password Complexity Requirements – **Verify** Password Minimum Age should be greater than X
- **Domain** Password Complexity Requirements – **Verify** Password History Count should be greater than X

Tech Savvy Productions

- **Domain** Password Complexity Requirements – **Verify** Password Reversible Encryption should be Disabled
- **Domain** Trust Availability – **Verify** each Trust status is OK
- **Domain** Trust Unconstrained TGTDelegation – **Verify** each Trust TGTDelegation is set to True
- **Domain** Kerberos Account Age – **Verify** Kerberos Last Password Change Should be less than 180 days
- **Domain** Groups: Account Operators – **Verify** Group is empty
- **Domain** Groups: Schema Admins – **Verify** Group is empty
- **Domain** User: Administrator – **Verify** Last Password Change should be less than 360 days or account disabled
- **Domain** DNS Forwarders – **Verify** DNS Forwarders are identical on all DNS nodes
- **Domain** DNS Scavenging Primary DNS Server – **Verify** DNS Scavenging is set to X days
- **Domain** DNS Scavenging Primary DNS Server – **Verify** DNS Scavenging State is set to True
- **Domain** DNS Scavenging Primary DNS Server – **Verify** DNS Scavenging Time is less than X days
- **Domain** DNS Zone Aging – **Verify** DNS Zone Aging is set
- **Domain** Well known folder – UsersContainer **Verify** folder is not at it's defaults.
- **Domain** Well known folder – ComputersContainer **Verify** folder is not at it's defaults.
- **Domain** Well known folder – DomainControllersContainer **Verify** folder is at it's defaults.
- **Domain** Well known folder – DeletedObjectsContainer **Verify** folder is at it's defaults.
- **Domain** Well known folder – SystemsContainer **Verify** folder is at it's defaults.
- **Domain** Well known folder – LostAndFoundContainer **Verify** folder is at it's defaults.
- **Domain** Well known folder – QuotasContainer **Verify** folder is at it's defaults.
- **Domain** Well known folder – ForeignSecurityPrincipalsContainer **Verify** folder is at it's defaults.

Tech Savvy Productions

- **Domain** Orphaned Foreign Security Principals – **Verify** there are no orphaned FSP objects.
- **Domain** Orphaned/Empty Organizational Units – **Verify** there are no orphaned Organizational Units
- **Domain** Group Policy Missing Permissions – **Verify** Authenticated Users/Domain Computers are on each and every Group Policy
- **Domain** DFSR Sysvol – **Verify** SYSVOL is DFSR
- **Domain Controller** Information – Is Enabled
- **Domain Controller** Information – Is Global Catalog
- **Domain Controller** Service Status – **Verify** all Services are running
- **Domain Controller** Service Status – **Verify** all Services are set to automatic startup
- **Domain Controller** Service Status (Print Spooler) – **Verify** Print Spooler Service is set to disabled
- **Domain Controller** Service Status (Print Spooler) – **Verify** Print Spooler Service is stopped
- **Domain Controller** Ping Connectivity – **Verify** DC is reachable
- **Domain Controller** Ports – **Verify** Following ports 53, 88, 135, 139, 389, 445, 464, 636, 3268, 3269, 9389 are open
- **Domain Controller** RDP Ports – **Verify** Following ports 3389 (RDP) is open
- **Domain Controller** RDP Security – **Verify** NLA is enabled
- **Domain Controller** LDAP Connectivity – **Verify** all LDAP Ports are open
- **Domain Controller** LDAP Connectivity – **Verify** all LDAP SSL Ports are open
- **Domain Controller** Windows Firewall – **Verify** windows firewall is enabled for all network cards
- **Domain Controller** Windows Remote Management – **Verify** Windows Remote Management identification requests are managed
- **Domain Controller** Resolves internal DNS queries – **Verify** DNS on DC resolves Internal DNS

Tech Savvy Productions

- **Domain Controller** Resolves external DNS queries – **Verify** DNS on DC resolves External DNS
- **Domain Controller** Name servers for primary domain zone **Verify** DNS Name servers for primary zone are identical
- **Domain Controller** Responds to PowerShell Queries **Verify** DC responds to PowerShell queries
- **Domain Controller** TimeSettings – **Verify** PDC should sync time to external source
- **Domain Controller** TimeSettings – **Verify** Non-PDC should sync time to PDC emulator
- **Domain Controller** TimeSettings – **Verify** Virtualized DCs should sync to hypervisor during boot time only
- **Domain Controller** Time Synchronization Internal – **Verify** Time Synchronization Difference to PDC less than X seconds
- **Domain Controller** Time Synchronization External – **Verify** Time Synchronization Difference to pool.ntp.org less than X seconds
- **Domain Controller** Disk Free – **Verify** OS partition Free space is at least X %
- **Domain Controller** Disk Free – **Verify** NTDS partition Free space is at least X %
- **Domain Controller** Operating System – **Verify** Windows Operating system is Windows 2012 or higher
- **Domain Controller** Windows Updates – **Verify** Last patch was installed less than 60 days ago
- **Domain Controller** SMB Protocols – **Verify** SMB v1 protocol is disabled
- **Domain Controller** SMB Protocols – **Verify** SMB v2 protocol is enabled
- **Domain Controller** SMB Shares – **Verify** default SMB shares NETLOGON/SYSVOL are visible
- **Domain Controller** DFSR AutoRecovery – **Verify** DFSR AutoRecovery is enabled
- **Domain Controller** Windows Roles and Features – **Verify** Windows Features for AD/DNS/File Services are enabled

And that is just a starting point, something to expand on. I've tried to pick different tests so I can see how easy for me is to add new tests without changing how **Testimo** works. The goal is to mostly spend time on building new tests without touching core too much. Of course, I may have

Tech Savvy Productions

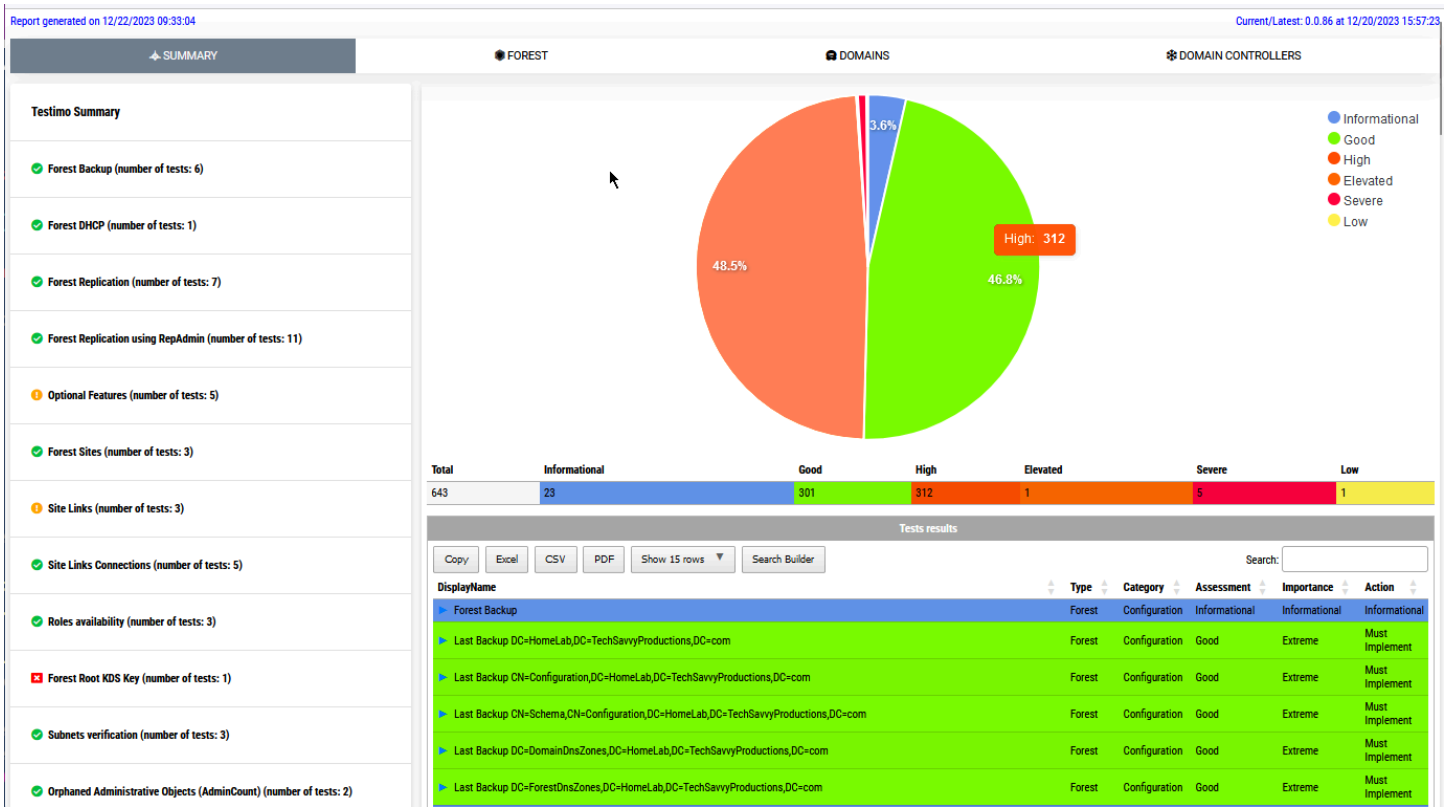
missed something or made an incorrect assumption, or even that in some cases, the test will fail for some reason. After all, this is an alpha product and something I've been developing for like three weeks on and off. Hopefully, with your help, I should be able to iron out the bugs, add new tests, and improve the quality of **Testimo** for different architectures. Maybe you can help out as well in other areas that need improvements? Feel free to reach out on GitHub/Twitter/Reddit/Discord. I'm in all those places.

Testimo - Running Tests

I gave you a little backstory, told the goals, shown some fancy feature list, but you're not here for that right? You want to see the action! Let's go straight for that! **Meet Testimo**! As with my earlier tests module again here, we have a single command that does **all the magic**.

1. Invoke-Testimo

What it does it executes what I call **a data source** (for example Get Last Backup Dates) and then it **verifies each data source** against **defined tests** (in this case checks whether last backup date is older than two days). Of course, the defaults are well, just defaults. Each **Active Directory**, each company has different needs.



As you can see above, it executed **420 tests**, of which **50 faile**d. But while **Invoke-Testimo** is excellent in delivering quick, visual assessment, it doesn't play well if you would want to act on it. That's why there is **ReturnResults** switch. The usage is simple:

1. Invoke-Testimo -ReturnResults

What changes? Well, you get the same visual output, but also you get **PowerShell object** when done. That means you can process results and act on them if you need to. It's **handy for automation** where you would execute this **daily/weekly** and if any of the tests you've defined returns false you can act on it.

Easy? It gives you an option to easily filter out on failed tests only, tests that relate to a particular **domain** or **domain controller**.

Testimo - Sending summary to email

Of course, you can do much more with that data. For example, if you were to use **Emailimo** (my go-to **PowerShell Module** to create beautiful, functional emails with **zero effort**) you could do something like this:

```
$Results = Invoke-Testimo -ReturnResults -ExcludeDomains 'ad.evotec.pl'

Email {
EmailHeader {
EmailFrom -Address 'myemail@evotec.pl'
EmailTo -Addresses "otheremail@evotec.pl"
EmailServer -Server 'smtp.office365' -UserName 'myemail@evotec.pl' -Password
'C:\Support\Important\Password-Evotec.txt' -PasswordAsSecure -PasswordFromFile -Port 587 -SSL
EmailOptions -Priority High -DeliveryNotifications Never
EmailSubject -Subject '[Reporting Evotec] Summary of Active Directory Tests'
}
EmailBody -FontFamily 'Calibri' -Size 15 {
EmailText -Text "Summary of Active Directory Tests" -Color None, Blue -LineBreak

EmailTable -DataTable $Results {
EmailTableCondition -ComparisonType 'string' -Name 'Status' -Operator eq -Value 'True' -BackgroundColor
Green -Color White -Inline -Row
EmailTableCondition -ComparisonType 'string' -Name 'Status' -Operator ne -Value 'True' -BackgroundColor
Red -Color White -Inline -Row
```


Tech Savvy Productions

```
} -HideFooter
}
} -AttachSelf -Supress $false
```

As you noticed, there's also a file attached. Since I've used **AttachSelf** switch, it attached an enhanced **HTML** version to email. When you open it up, you get a bit more options to play with.

You can export your content to Excel, filter things out, and play as needed. Cool right? **Emailimo** will be integrated into **Testimo** so that you can fill in your email address and some basic settings into configuration and whenever you run it with **Invoke-Testimo -Email** switch it would use email information to give you report as needed. Surely that's something I would love to get feedback on thou. This is planned for the next release.

Testimo - Advanced Configuration

As I mentioned it before every **Active Directory** is different so that rules can be different. While for **ad-hoc** tests that you do to asses health check of **Active Directory** it's fine to visually skip false-positives, using it daily with red all over the place isn't the best way to go. That's why **Testimo** allows you to export/edit configuration in three ways. To **file/JSON**:

1. Get-TestimoConfiguration -FilePath $PSScriptRoot\Configuration\TestimoConfiguration.json

Which delivers something that looks like **configuration** below where you can change your settings as you need them.

To **JSON directly** (in case you want to process data however you want to handle it):

1. Get-TestimoConfiguration -AsJson

And finally output to **hashtable** which you can edit while being in **PowerShell**:

1. $OutputOrderedDictionary = Get-TestimoConfiguration
2. $OutputOrderedDictionary.Forest.OptionalFeatures.Tests.RecycleBinEnabled.Enable = $false
3. $OutputOrderedDictionary.Forest.OptionalFeatures.Tests.LapsAvailable.Enable = $true

Tech Savvy Productions

4. $OutputOrderedDictionary.Forest.OptionalFeatures.Tests.LapsAvailable.Parameters.ExpectedValue = $false

How you intend to use it, it's up to you. **Exporting configuration** and modifying it doesn't do much. You still need to **import that configuration** back to **Testimo**.

```
$ConfigurationFile = "$PSScriptRoot\Configuration\TestimoConfiguration.json"

$Sources = @(
#'ForestFSMORoles'
'ForestOptionalFeatures'
'ForestBackup'
#'ForestOrphanedAdmins'
'DomainPasswordComplexity'
#'DomainKerberosAccountAge'
#'DomainDNSScavengingForPrimaryDNSServer'
'DCWindowsUpdates'
)

$TestResults = Invoke-Testimo -ReturnResults -ExcludeDomains 'ad.evotec.pl' -ExtendedResults
-Configuration $ConfigurationFile -Sources $Sources
$TestResults | Format-Table -AutoSize *
```

I hope that didn't scare you with that code above right? As you can see above I am loading configuration from **JSON** file, I've also decided that I want to partially limit the number of tests I want to run (even thou configuration has it all enabled) using **Sources** parameter, which means **only four sources** will be used in next run. I've also decided to skip one of my broken domains because it takes a lot more time to scan a broken domain (timeouts on every single test take time). Finally, I'm using **ReturnResults** but also with another switch **ExtendedResults**. This will give you a bit of individual output that I'll describe a bit later on. Alternatively, the Configuration parameter also takes **Hashtables** as input.

1. $OutputOrderedDictionary = Get-TestimoConfiguration
2. $OutputOrderedDictionary.Forest.OptionalFeatures.Tests.RecycleBinEnabled.Enable = $false
3. $OutputOrderedDictionary.Forest.OptionalFeatures.Tests.LapsAvailable.Enable = $true
4. $OutputOrderedDictionary.Forest.OptionalFeatures.Tests.LapsAvailable.Parameters.ExpectedValue = $false
5.
6. $Sources = @(
7. #'ForestFSMORoles'

Tech Savvy Productions

8. 'ForestOptionalFeatures'
9. 'ForestBackup'
10. #'ForestOrphanedAdmins'
11. 'DomainPasswordComplexity'
12. #'DomainKerberosAccountAge'
13. #'DomainDNSScavengingForPrimaryDNSServer'
14. 'DCWindowsUpdates'
15.)
16.
17. $TestResults = Invoke-Testimo -ReturnResults -ExcludeDomains 'ad.evotec.pl' -ExtendedResults -Sources $Sources -Configuration $OutputOrderedDictionary
18. $TestResults | Format-Table -AutoSize *

It's up to you which one you prefer better. Keep in mind that what **Sources** do is they limit tests to a defined list, which is useful. However, there's also **ExcludeSources** which you can use if you want to skip one or two tests but leave everything else.

Testimo - Extended Report

As mentioned earlier next to **ReturnResults**, there's also **ExtendedResults** switch. When you run **Testimo** with that switch, the result is not impressive. To not spend the next 10 minutes waiting for all tests to finish, I'm going to limit the number of sources and use two types of tests, along with two mentioned switches.

1. Invoke-Testimo -Sources ForestRoles,DomainRoles -ReturnResults -ExtendedResults

See, the result is as usual displayed to screen, and additionally, **PowerShell object** is returned. This object is unique and something that is work in progress that I intend to use to build an excellent looking **Overview** of it. I'm returning it so you can take it and make something of your own if you want to. Below is a code I'm using, to create **Overview** of all tests that happened (that's PSWriteHTML in action).

1. $Sources = @(
2. 'ForestRoles'

Tech Savvy Productions

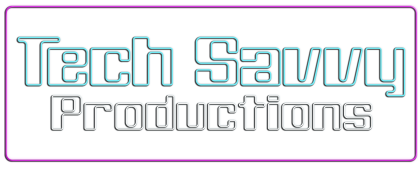
3. 'ForestOptionalFeatures'
4. 'ForestOrphanedAdmins'
5. 'DomainPasswordComplexity'
6. 'DomainKerberosAccountAge'
7. 'DomainDNSScavengingForPrimaryDNSServer'
8. 'DomainSysVolDFSR'
9. 'DCRDPSecurity'
10. 'DCSMBShares'
11. 'DomainGroupPolicyMissingPermissions'
12. 'DCWindowsRolesAndFeatures'
13. 'DCNTDSParameters'
14. 'DCInformation'
15. 'ForestReplicationStatus'
16.)
17.
18. $TestResults = Invoke-Testimo -ReturnResults -ExtendedResults -Sources $Sources #-ExcludeDomains 'ad.evotec.pl' #-ExcludeDomainControllers $ExludeDomainControllers
19.
20. New-HTML -FilePath $PSScriptRoot\Output\TestimoSummary.html -UseCssLinks -UseJavaScriptLinks {
21. [Array] $PassedTests = $TestResults['Results'] | Where-Object { $_.Status -eq $true }
22. [Array] $FailedTests = $TestResults['Results'] | Where-Object { $_.Status -ne $true }
23. New-HTMLTab -Name 'Summary' -IconBrands galactic-senate {
24. New-HTMLSection -HeaderText "Tests results" -HeaderBackGroundColor DarkGray {
25. New-HTMLPanel {
26. New-HTMLChart {

Tech Savvy Productions

```
27. New-ChartPie -Name 'Passed' -Value ($TestResults['Summary'].Passed) -Color
    ForestGreen
28. New-ChartPie -Name 'Failed' -Value ($TestResults['Summary'].Failed) -Color OrangeRed
29. New-ChartPie -Name 'Failed' -Value ($TestResults['Summary'].Skipped) -Color
    LightBlue
30. }
31. New-HTMLTable -DataTable $TestResults['Summary'] -HideFooter -DisableSearch {
32. New-HTMLTableContent -ColumnName 'Passed' -BackGroundColor ForestGreen -Color
    White
33. New-HTMLTableContent -ColumnName 'Failed' -BackGroundColor OrangeRed -Color
    White
34. New-HTMLTableContent -ColumnName 'Skipped' -BackGroundColor LightBlue -Color
    White
35. }
36. }
37. New-HTMLPanel {
38. New-HTMLTable -DataTable $TestResults['Results'] {
39. New-HTMLTableCondition -Name 'Status' -Value $true -Color Green -Row
40. New-HTMLTableCondition -Name 'Status' -Value $false -Color Red -Row
41. }
42. }
43. }
44. }
45. New-HTMLTab -Name 'Forest' -IconBrands first-order {
46. foreach ($Source in $TestResults['Forest']['Tests'].Keys) {
47.
48. $Name = $TestResults['Forest']['Tests'][$Source]['Name']
49. $Data = $TestResults['Forest']['Tests'][$Source]['Data']
```


Tech Savvy Productions

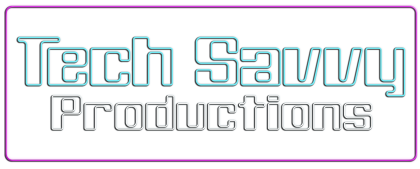
```
$SourceCode = $TestResults['Forest']['Tests'][$Source]['SourceCode']
$Results = $TestResults['Forest']['Tests'][$Source]['Results']
#$Details = $TestResults['Forest']['Tests'][$Source]['Details']
[Array] $PassedTestsSingular = $TestResults['Forest']['Tests'][$Source]['Results'] |
    Where-Object { $_.Status -eq $true }
[Array] $FailedTestsSingular = $TestResults['Forest']['Tests'][$Source]['Results'] |
    Where-Object { $_.Status -ne $true }

New-HTMLSection -HeaderText $Name -HeaderBackGroundColor DarkGray
    -CanCollapse {
New-HTMLContainer {
New-HTMLPanel {
New-HTMLChart {
New-ChartPie -Name 'Passed' -Value ($PassedTestsSingular.Count) -Color ForestGreen
New-ChartPie -Name 'Failed' -Value ($FailedTestsSingular.Count) -Color OrangeRed
}
New-HTMLCodeBlock -Code $SourceCode -Style 'PowerShell' -Theme enlighter
}
}
New-HTMLContainer {
New-HTMLPanel {
New-HTMLTable -DataTable $Data
New-HTMLTable -DataTable $Results {
New-HTMLTableCondition -Name 'Status' -Value $true -Color Green -Row
New-HTMLTableCondition -Name 'Status' -Value $false -Color Red -Row
}
}
```

```
74.}
75.}
76.}
77.}
78.
79.foreach ($Domain in $TestResults['Domains'].Keys) {
80.New-HTMLTab -Name "Domain $Domain" -IconBrands deskpro {
81.foreach ($Source in $TestResults['Domains'][$Domain]['Tests'].Keys) {
82.$Name = $TestResults['Domains'][$Domain]['Tests'][$Source]['Name']
83.$Data = $TestResults['Domains'][$Domain]['Tests'][$Source]['Data']
84.$SourceCode = $TestResults['Domains'][$Domain]['Tests'][$Source]['SourceCode']
85.$Results = $TestResults['Domains'][$Domain]['Tests'][$Source]['Results']
86.# $Details = $TestResults['Domains'][$Domain]['Tests'][$Source]['Details']
87.[Array] $PassedTestsSingular =
    $TestResults['Domains'][$Domain]['Tests'][$Source]['Results'] | Where-Object {
    $_.Status -eq $true }
88.[Array] $FailedTestsSingular =
    $TestResults['Domains'][$Domain]['Tests'][$Source]['Results'] | Where-Object {
    $_.Status -ne $true }
89.
90.New-HTMLSection -HeaderText $Name -HeaderBackGroundColor DarkGray
    -CanCollapse {
91.New-HTMLContainer {
92.New-HTMLPanel {
93.New-HTMLChart {
94.New-ChartPie -Name 'Passed' -Value ($PassedTestsSingular.Count) -Color ForestGreen
95.New-ChartPie -Name 'Failed' -Value ($FailedTestsSingular.Count) -Color OrangeRed
96.}
```


Tech Savvy Productions

```
97.New-HTMLCodeBlock -Code $SourceCode -Style 'PowerShell' -Theme enlighter
98.}
99.}
100.  New-HTMLContainer {
101.  New-HTMLPanel {
102.  New-HTMLTable -DataTable $Data
103.  New-HTMLTable -DataTable $Results {
104.  New-HTMLTableCondition -Name 'Status' -Value $true -Color Green -Row
105.  New-HTMLTableCondition -Name 'Status' -Value $false -Color Red -Row
106.  }
107.  }
108.  }
109.  }
110.  }
111.  foreach ($DC in $TestResults['Domains'][$Domain]['DomainControllers'].Keys) {
112.  New-HTMLSection -HeaderText "Domain Controller - $DC"
    -HeaderBackGroundColor DarkSlateGray -CanCollapse {
113.  New-HTMLContainer {
114.  foreach ($Source in
    $TestResults['Domains'][$Domain]['DomainControllers'][$DC]['Tests'].Keys) {
115.  $Name =
    $TestResults['Domains'][$Domain]['DomainControllers'][$DC]['Tests'][$Source]['Name']
116.  $Data =
    $TestResults['Domains'][$Domain]['DomainControllers'][$DC]['Tests'][$Source]['Data']
117.  $SourceCode =
    $TestResults['Domains'][$Domain]['DomainControllers'][$DC]['Tests'][$Source]['Source
    Code']
```

```
118.  $Results =
      $TestResults['Domains'][$Domain]['DomainControllers'][$DC]['Tests'][$Source]['Results
      ']
119.  #$Details =
      $TestResults['Domains'][$Domain]['DomainControllers'][$DC]['Tests'][$Source]['Details'
      ]
120.  [Array] $PassedTestsSingular =
      $TestResults['Domains'][$Domain]['DomainControllers'][$DC]['Tests'][$Source]['Results
      '] | Where-Object { $_.Status -eq $true }
121.  [Array] $FailedTestsSingular =
      $TestResults['Domains'][$Domain]['DomainControllers'][$DC]['Tests'][$Source]['Results
      '] | Where-Object { $_.Status -ne $true }
122.
123.  New-HTMLSection -HeaderText $Name -HeaderBackGroundColor DarkGray {
124.  New-HTMLContainer {
125.  New-HTMLPanel {
126.  New-HTMLChart {
127.  New-ChartPie -Name 'Passed' -Value ($PassedTestsSingular.Count) -Color
      ForestGreen
128.  New-ChartPie -Name 'Failed' -Value ($FailedTestsSingular.Count) -Color OrangeRed
129.  }
130.  New-HTMLCodeBlock -Code $SourceCode -Style 'PowerShell' -Theme enlighter
131.  }
132.  }
133.  New-HTMLContainer {
134.  New-HTMLPanel {
135.  New-HTMLTable -DataTable $Data
136.  New-HTMLTable -DataTable $Results {
137.  New-HTMLTableCondition -Name 'Status' -Value $true -Color Green -Row
```


Tech Savvy Productions

```
138.    New-HTMLTableCondition -Name 'Status' -Value $false -Color Red -Row
139.    }
140.    }
141.    }
142.    }
143.    }
144.    }
145.    }
146.    }
147.    }
148.    }
149.    } -ShowHTML
```

Which will give you the following **HTML** document

As you can see above, there are some tabs. The **first tab** has a **summary of all tests**, and I plan to expand it with some charts and more data that may be useful for quick assessment. Following pages give you a **detailed overview of all tests** for Forest, and each Domain separately. Please notice that on the left side, just under pie chart, it shows you a PowerShell command that is executed to get the data. On the right side, there is output from that command, and just below it tests results. The idea is that if something is wrong, you can see more information on the problem straight away or you can take the matter in your own hands and use the provided **PowerShell** commands and see for yourself. Of course, you may need to replace place holder variables with your own data. I may make it more comfortable in the next versions by filling that for you. This is just a very early concept of course, so I will make sure this gets some "I want to look good" treatment. I look forward to hearing your feedback on this. Since I know whatever I will build here, may not be not enough – all that data I used to create that output internally is available by using **ReturnResults**, **ExtendedResults** switches. The report from above is also available with just this little command. Keep in mind I'm limiting **sources**, so I don't spend 10 minutes waiting for it to execute everything. You can skip sources parameter and get full test output to **HTML** for all your Forest/Domain data.

Tech Savvy Productions

```
Import-Module .\Testimo.psd1 -Force #-Verbose

$Sources = @(
'ForestRoles'
'ForestOptionalFeatures'
'ForestOrphanedAdmins'
'DomainPasswordComplexity'
'DomainKerberosAccountAge'
#'DomainDNSScavengingForPrimaryDNSServer'
#'DomainSysVolDFSR'
#'DCRDPSecurity'
'DCSMBShares'
'DomainGroupPolicyMissingPermissions'
#'DCWindowsRolesAndFeatures'
#'DCNTDSParameters'
#'DCInformation'
'ForestReplicationStatus'
)

Invoke-Testimo -Sources $Sources -ExcludeDomains 'ad.evotec.pl' -ShowReport
```

The report is available online for you to view – **Testimo Example Report**.

Testimo - Test Sources

While you may be wondering what **Sources** are available, I've decided to simplify this for you. Just run a command, and you get all the available sources. It's important because the same idea will apply to export new configuration. Keep in mind that this project is not yet written in stone and not everything is yet decided so some things may change, some typos will get fixed, or definitions changed. I'll try to make it as painless as possible thou.

```
Get-TestimoSources
```


Tech Savvy Productions

- DCNTDSParameters
- DCDFSRAutoRecovery
- DCDiskSpace
- DCDnsNameServes
- DCDnsResolveExternal
- DCDnsResolveInternal
- DCLDAP
- DCOperatingSystem
- DCInformation
- DCPingable
- DCPorts
- DCRDPPorts
- DCRDPSecurity
- DCServices
- DCSMBProtocols
- DCSMBShares
- DCTimeSettings
- DCTimeSynchronizationExternal
- DCTimeSynchronizationInternal
- DCWindowsFirewall
- DCWindowsRemoteManagement
- DCWindowsRolesAndFeatures
- DCWindowsUpdates
- DomainDNSForwaders
- DomainDNSScavengingForPrimaryDNSServer
- DomainDnsZonesAging

Tech Savvy Productions

- DomainEmptyOrganizationalUnits
- DomainGroupPolicyMissingPermissions
- DomainKerberosAccountAge
- DomainOrphanedForeignSecurityPrincipals
- DomainPasswordComplexity
- DomainRoles
- DomainSecurityGroupsAccountOperators
- DomainSecurityGroupsSchemaAdmins
- DomainSecurityUsersAcccountAdministrator
- DomainSysVolDFSR
- DomainTrusts
- DomainWellKnownFolders
- ForestBackup
- ForestOptionalFeatures
- ForestOrphanedAdmins
- ForestReplication # This uses newer approach
- ForestReplicationStatus # This uses Repadmin
- ForestRoles
- ForestSiteLinks
- ForestSiteLinksConnections
- ForestSites
- ForestTombstoneLifetime

Of course, when you are using **Sources** parameter, it will autocomplete as well. But sometimes it's faster to build array without using autocomplete. It's essential to understand that each **Source** can have multiple tests assigned to it. For example, you could get a list of all **Domain Controllers** and their availability. You could then do multiple tests checking different things based on one source. I have defined some criteria on each source, but it's possible to expand on it and establish more tests to the same source.

Tech Savvy Productions

Testimo Requirements

Testimo to work requires **PowerShell 5.1**. It can be installed on **Windows 2008 R2** however from my tests it misses a lot of commands I use for testing DNS or other parts of the system. Since **2008 R2 extended support** ends in just a few months, it seems a bit unnecessary to add support for it. However, I will consider this if there will be interest to support it. A bigger problem is I don't have **Windows 2008R2** to test with. Anything above **Windows 2012** should work correctly. Feel free to let me know if it doesn't, and we can figure this out. If you're running this from **Windows 10** machine, please make sure you have correct permissions to run it against your **Active Directory** and that you have **RSAT** installed. I'm doing most of the tests using **Windows 10 1903**.

Installing & Updating Testimo

1. Install-Module Testimo

For easy use and installation, all modules are available from **PowerShellGallery**. Installing it is as easy as it gets. Keep in mind that when you install **Testimo**, you get **PSWinDocumentation.AD**, **PSWinDocumentation.DNS**, **ADEssentials**, **PSSharedGoods**, **PSWriteColor**, **Connectimo**, **PSWriteHTML** and **Emailimo** installed by default, so you don't have to install it separately. You may be wondering why so many modules? Well, I already have some stuff written for different purposes, and while I could get the stuff I need from different modules, I wrote and keep everything integrated, it would kill my productivity.

Sometimes you may need additional steps

1. Install-Module Testimo -Force -AllowClobber

Since Testimo is available as **PowerShell** module new versions are also published there. When a new version is out, you can run

1. Update-Module Testimo

Using **Force** and **AllowClobber** is optional and should be only used if you experience issues in installation. For example, if you run **Install-Module** with **Force** switch while **Testimo** is already installed, it will overwrite/install the newest version, but it will also redownload and overwrite any dependencies it has. **AllowClobber**, on the other hand, is useful when I decide to move functions between modules. I may decide that function X should be in module Y instead of module Z. When you already have function X in module Z, and you move it in next version to module Y PowerShell will detect that there is already command with that name on disk and it will complain. That's where **AllowClobber** comes in where it tells **PowerShell** to ignore this

Tech Savvy Productions

problem for now. Since both module Z and Y will get updated function will be in only one of them, and everything will start to work.

Making Testimo Portable

Since Testimo aims to be useful also in scenarios where you don't want to install **Testimo**, and it's dependencies to **PowerShell Modules** path I've created a simple function that downloads Testimo and it's dependencies from **PowerShellGallery**, and finally uses **Import-Module** to import modules into memory so that you can use it as you like. I've written a short **blog** about it with code you can use to get that up and running.

1. Initialize-ModulePortable -Name 'Testimo' -Path $Env:UserProfile\Desktop\TestimoPortable -Download

After that **Invoke-Testimo** and all its features are available within-session, you're using it. Of course, you don't have to use the Download switch all the time, but only when you update **Testimo** and it's dependencies to a newer version. If you don't use the download switch all it does is preload all modules from a given path, and that's it.

Uninstalling & Removing Testimo

Since you can install it, you can also uninstall it. Of course it's not enought to just uninstall Testimo because all other dependencies will stay there. If you're not using any of those other modules that I've written, feel free to uninstall all of them.

```
1.$Modules = @('Testimo',
  'PSWinDocumentation.AD','PSWinDocumentation.DNS','ADEssent
  ials', 'PSSharedGoods','PSWriteColor', 'Connectimo',
  'DSInternals' )
2.foreach ($Module in $Modules) {
3.Uninstall-Module $Module -Force -AllVersions
4.}
```

Useful tools for reporting/emailing

Since **Reporting**/**Emailing** functionality of **Testimo** is not ready and can change feel free to explore what is possible with PSWriteHTML and Emailimo. Below is a bunch of links describing those modules doing different actions. If you find there something that you think would look cool in Testimo reporting, let me know.

- Meet Statusimo – PowerShell generated Status Page

- Meet Dashimo – PowerShell Generated Dashboard
- Dashimo – Easy Table Conditional Formatting and more
- Out-HtmlView – HTML alternative to Out-GridView
- Meet Emailimo – New way to send pretty emails with PowerShell
- All your HTML Tables are belong to us
- Sending HTML emails with PowerShell and zero HTML knowledge required
- Dashimo (PSWriteHTML) – Charting, Icons and few other changes
- Working with HTML in PowerShell just got better
- Comparing two or more objets visually in PowerShell (cross-platform)

Development Features Requests Bugs

Testimo and all other modules are open source and are available on GitHub. If you feel you can help with just about anything – from fixing typos to adding more tests – you're very welcome!

- Testimo – Engine mostly, this is where main development happens
- PSSharedGoods – my "glue" PowerShell module that holds a lot of functions that are used by my modules
- PSWriteHTML – responsible for HTML reporting
- ADEssentials – some commands I used are based here
- PSWinDocumentation.AD – while it's a module to build AD Documentation I use some internal commands to build tests
- PSWinDocumentation.DNS – same as above but for DNS (not ready for documentation, useful for tests – work in progress)
- Emailimo – module for sending/building HTML based emails.

Have fun!

active directoryaddhcpdnshealth checkspowershellsecurity checkstestimo

Przemyslaw Klys / About Author

System Architect with over 14 years of experience in the IT field. Skilled, among others, in Active Directory, Microsoft Exchange and Office 365. Profoundly interested in PowerShell. Software geek.

Tech Savvy Productions

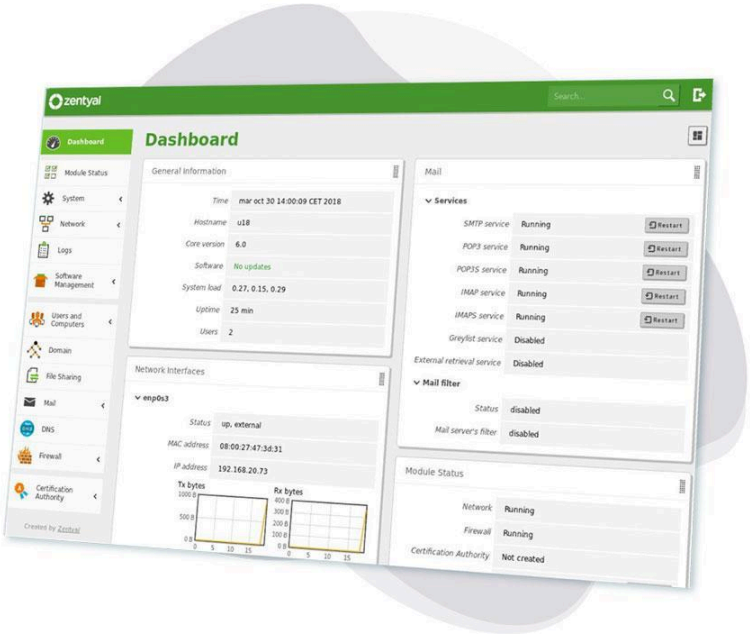
https://zentyal.com/features/

Zentyal: Active Directory using Linux

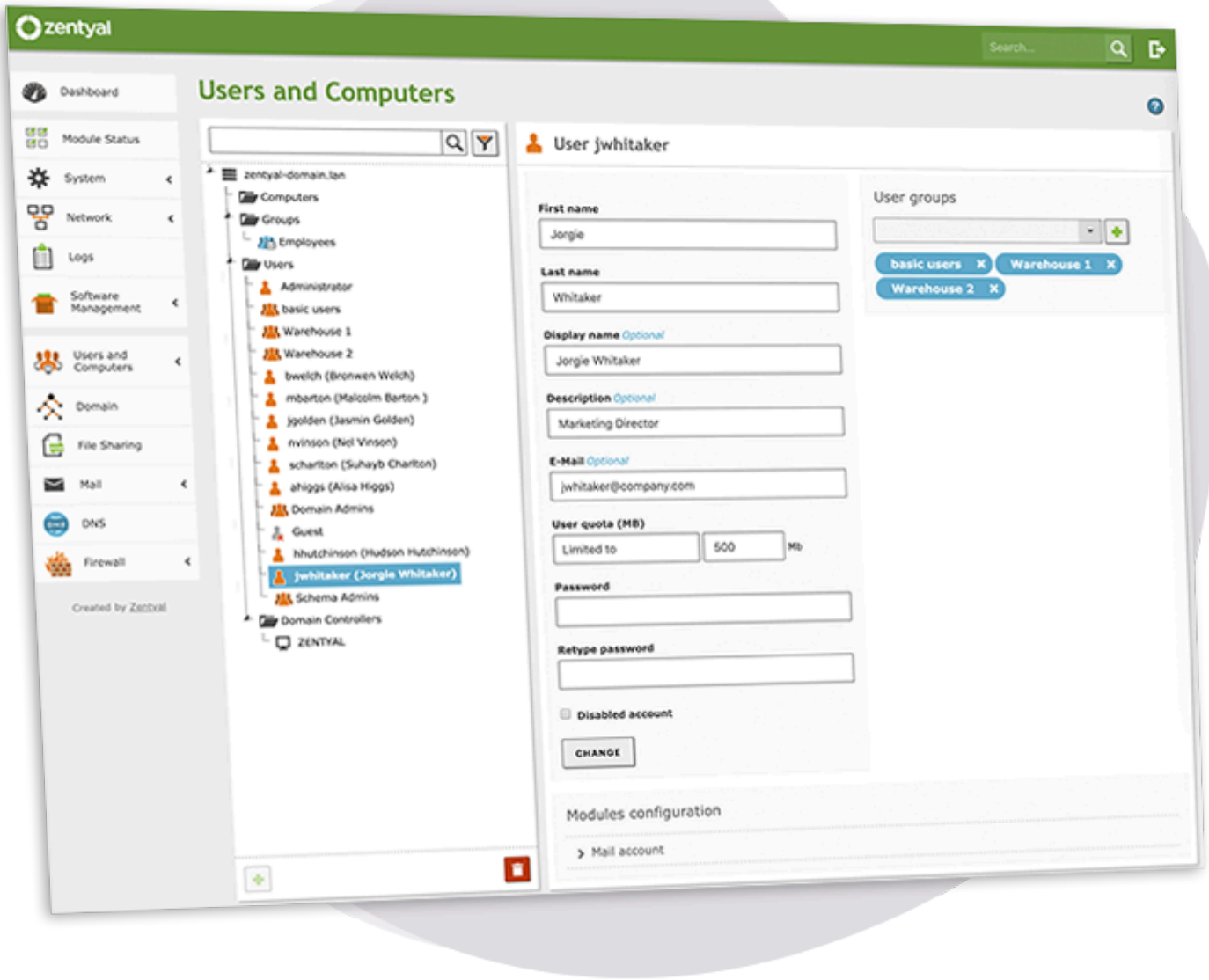
Ease the management of your IT infrastructure

Manage your Windows® users as you are used to, without the associated cost. Plus unify network infrastructure management to get to focus on the business critical IT of your organization.

Tech Savvy Productions



Tech Savvy Productions



Domain & Directory Server

Replace your MS Active Directory® seamlessly

- Set up as a stand-alone server or additional DC of a Windows domain
- Manage GPOs with RSAT
- Inexpensive, doesn't require CALs

Zentyal offers an easy to use Windows Server® alternative. It comes with native compatibility with Microsoft Active Directory® allowing you to join Windows® clients to the domain and manage them easily, causing no disruption to your users.

Tech Savvy Productions

See all domain & directory features

Mail Server

Dedicated mail server

- Integrates modern webmail and ActiveSync
- Unlimited virtual mail domains
- User management via Zentyal or Microsoft® Active Directory

Zentyal includes the industry-standard SMTP and POP3/IMAP mail servers built upon the most established technologies and protocols. Gives you the opportunity to deploy Zentyal as a mail server, domain & directory server with mail or all-in one server.

See all mail features

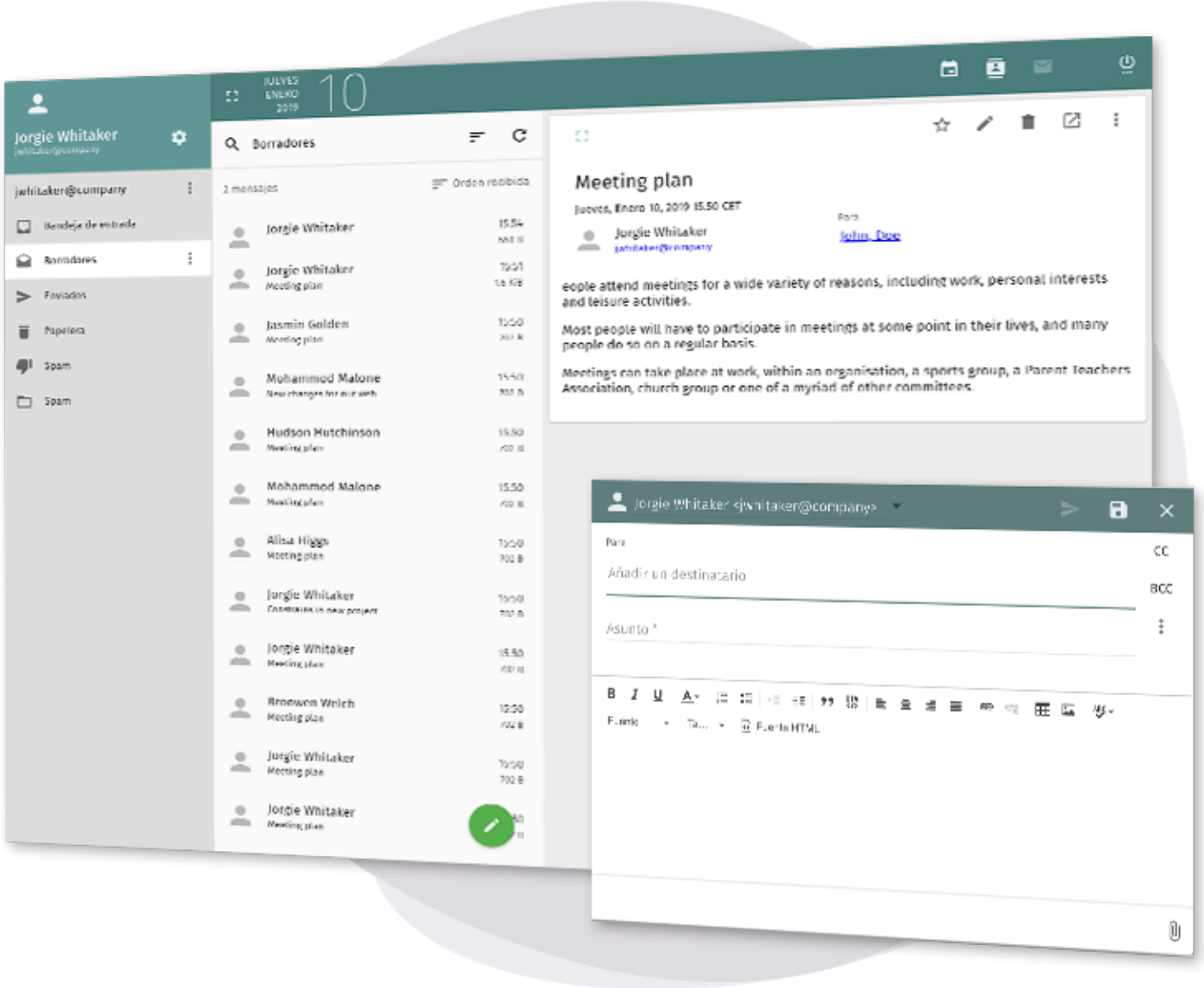
He has shown you, O mortal, what is good.

And what does the Lord require of you?

To act justly and to love mercy

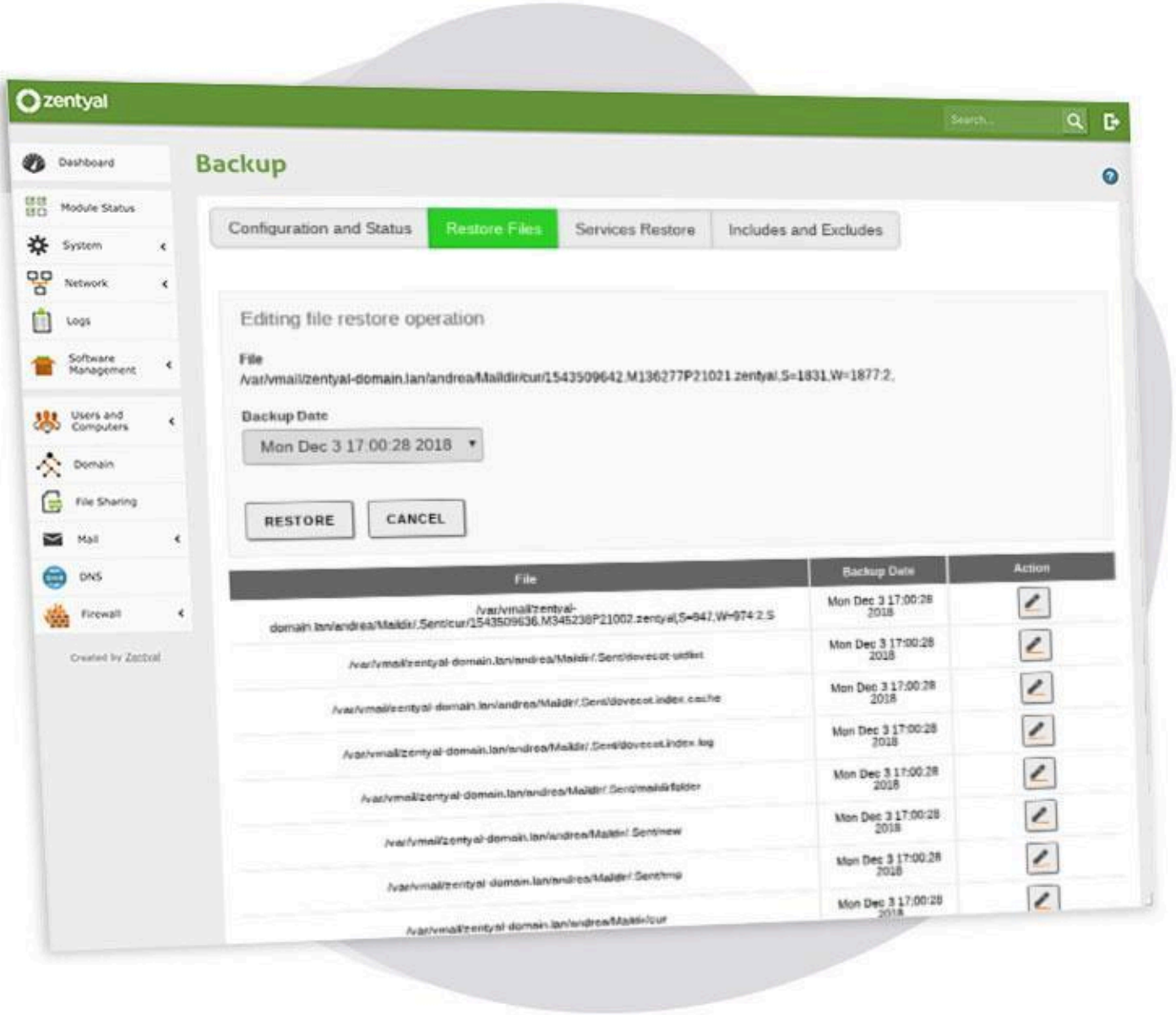
and to walk humbly with your God.

Tech Savvy Productions



Love is patient, love is kind. It does not envy, it does not boast, it is not proud. It does not dishonor others, it is not self-seeking, it is not easily angered, it keeps no record of wrongs. Love does not delight in evil but rejoices with the truth. It always protects, always trusts, always hopes, always perseveres.

Tech Savvy Productions



Gateway & Infrastructure Server

Secure Internet access and unify your network infrastructure

- Backup
- User authentication in HTTP Proxy
- Domain-based HTTPS web pages block

Zentyal allows you to unify and easily manage all the basic network infrastructure services and to offer reliable and secure Internet access. It covers services from DNS/DHCP, CA, VPNs and backup to gateway, firewall and HTTP proxy to mention few.

Tech Savvy Productions

See all gateway and infrastructure features

Technical Support

Expert assistance when you need it

- Covers all the Zentyal Server features
- Provided by the Official Zentyal Support Team
- Best-effort support for migrations from Windows Server

Not quite sure if you have the necessary knowledge to set-up and maintain Zentyal Server in-house? Count on the Zentyal Support Team – They have extensive experience in providing support for commercial Zentyal deployments.

See all support features

Tech Savvy Productions

Pricing

Perpetual licenses and optional support

Purchase perpetual Zentyal Server License according to your business size, starting from 195€/server (Micro License, <25 users). We have server-based pricing, no user or device CALs are required. Optional Support Subscriptions are also available.

For further pricing information, simply request an offer.

Tech Savvy Productions

Request an offer

Full feature list

Latest release: **Zentyal 7.0**
based on Ubuntu Server 20.04 LTS

Directory & Domain	• Central domain and directory management • Users, Security groups, Distribution lists, Contacts • Multiple Organization Units (OUs), Group Policy Objects (GPOs) • NETLOGON scripts, Roaming profiles • Single Sign-On (SSO) authentication • Supported OS: Windows® XP/Vista/7/8/10 • File sharing in Windows® environments (CIFS) • Users and Groups access and modification permissions (ACLs) • Management of user profile pictures • Importing/exporting of users and groups • Audit mode on/off (file server) • Integrated software: Samba
Mail	• Supported protocols: SMTP, POP3, IMAP, CalDAV, CardDAV, SIEVE • Supported clients: Mozilla Thunderbird® • Webmail • Synchronization to mobile devices via ActiveSync • Multiple virtual mail domains • Single Sign-On (SSO) authentication • Management via Zentyal or Microsoft® Active Directory • Antivirus & Mail filter • Integrated software: Postfix, Dovecot, Fetchmail, Sieve, SOGo, SOGo ActiveSync, Amavis, ClamAV, SpamAssasin

Tech Savvy Productions

Gateway	• Network configuration • Routing • Gateway • Firewall • Network authentication service (RADIUS) • HTTP Proxy • IDS/IPS • User authentication in HTTP Proxy • Domain-based HTTPS web pages block • Integrated software: Iproute2, Netfilter, Squid, Suricata, FreeRADIUS • DHCP and DNS server • NTP server • Certification Authority (CA)
Infrastructure	• Virtualization Manager • Virtual Private Networks (VPNs) • Backup • Instant Messaging (IM) service • FTP Server • IPSec/L2TP • Antivirus on-access scan • Integrated software: BIND, ISC DHCP Software, ntpd, OpenSSL, OpenVPN, ejabbered, Libvirt/KVM, Duplicity, vsftpd, Libreswan
Maintenance	• Real-time alerts • Daily reports • Kernel management

Tech Savvy Productions

Support & Updates	• Software and security updates • Software upgrades • Access to the knowledge base • Technical support

Now all has been heard;
here is the conclusion of the matter:
Fear God and keep his commandments,
for this is the duty of all mankind.
For God will bring every deed into judgment,
including every hidden thing,
whether it is good or evil.

Tech Savvy Productions


Tech Savvy
Productions