Dumping Active Directory Domain Info – in Go!

I’ve used NetSPI PowerShell tools and the PowerView toolset to dump information from Active Directory during almost every internal penetration test I’ve done. These tools are a great starting point for gaining insight into an Active Directory environment. Go seems to be gaining popularity for its performance and scalability, so I tried to replicate some of the functionality in my favorite PowerShell tools. **goddi** (go dump domain info) dumps domain users, groups, domain controllers, and more in CSV output. And it runs on Windows and Linux!

Before going any further, I want to thank Scott Sutherland (@**_nullbind**) for his help and mentorship. This work is based off of internal tools he created and none of it would be possible without him! This tool is also based on work from Antti Rantasaari, Eric Gruber (@egru), Will Schroeder (@harmj0y), and the PowerView authors.

So Why Go?

Go is fast and supports cross platform compilation. During testing, goddi managed to cut execution time down to a matter of seconds when compared to its PowerShell counterparts. Go binaries can also be built for Windows, Linux, and MacOS all on the same system. The full list of OS and architecture combinations are listed in the go GitHub repo. At the time of this blog’s release, goddi has been tested on Windows (10 and 8.1) and Kali Linux.

That isn’t to say that there aren’t any drawbacks with a Go implementation. The Microsoft ADSI API is much more flexible to work with, especially when creating LDAP queries to run under the current user’s security context. goddi requires domain credentials to be explicitly provided on the command line. This can be especially annoying in scenarios where a user’s credentials may not be known. If you get access to a box with local Administrator, but don’t have domain credentials yet, you can run PSExec to get local system. With local system, you can check if you have domain user privileges and then run PowerShell in this current context without domain credentials. This functionality is on the roadmap for future development.

Features

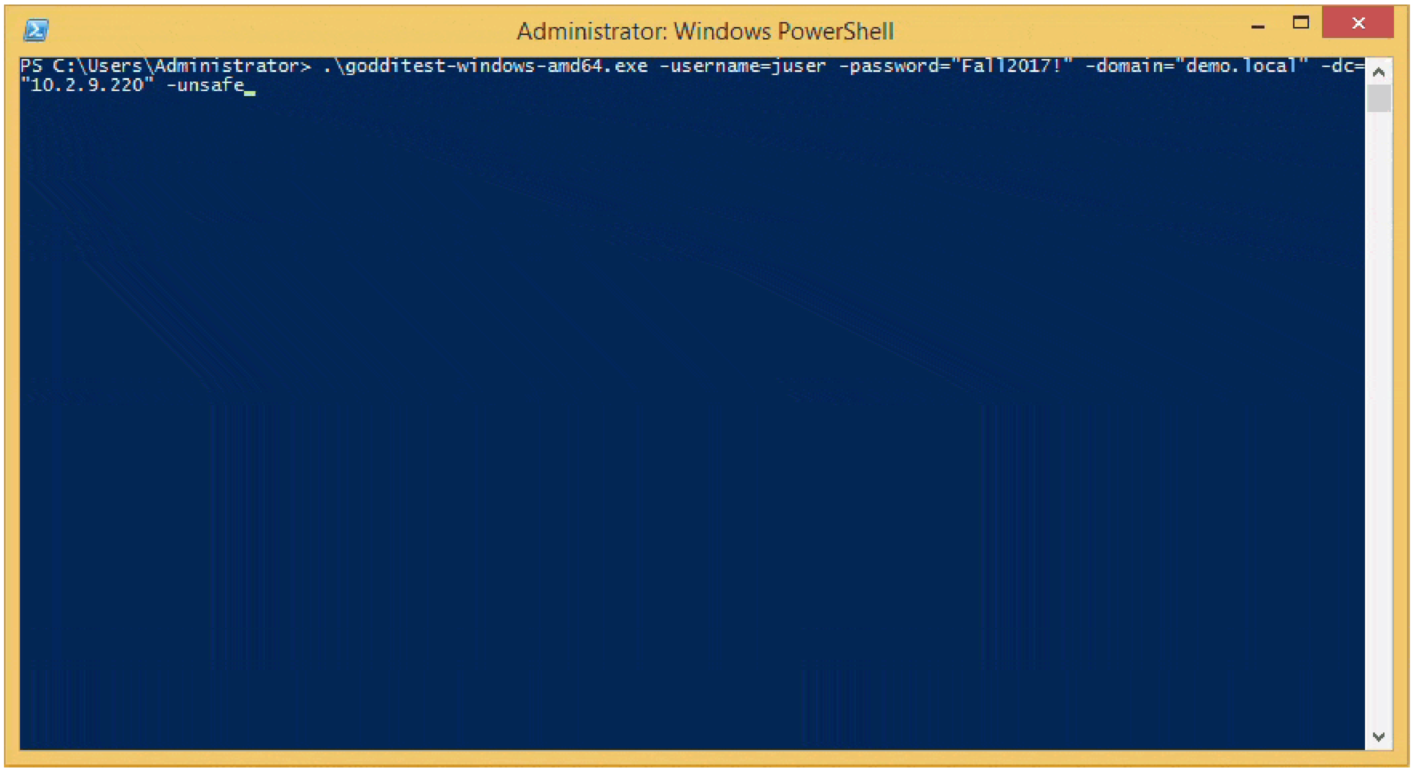
Check out the GitHub repo for an up to date list of features. goddi dumps…

- Domain users
- Users in privileged user groups (DA, EA, FA)
- Users with passwords not set to expire
- User accounts that have been locked or disabled
- Machine accounts with passwords older than 45 days
- Domain Computers
- Domain Controllers
- Sites and Subnets
- SPNs
- Trusted domain relationships
- Domain Groups

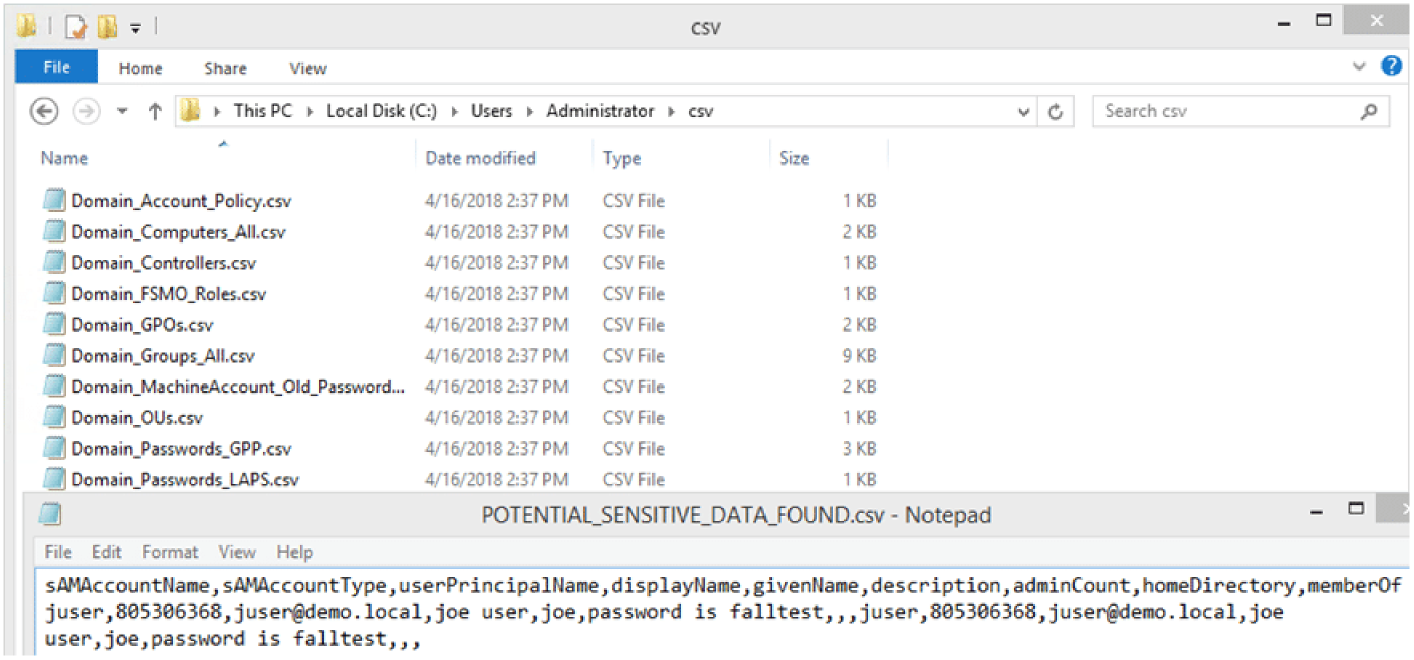
- Domain OUs
- Domain Account Policy
- Domain deligation users
- Domain GPOs
- Domain FSMO roles
- LAPS passwords
- GPP passwords

Run goddi with the example command below. The CSV output is dumped in the “csv” folder in the current working directory.

```
goddi-windows-amd64.exe -username=juser -password="Fall2017!" -
domain="demo.local" -dc="10.2.9.220" -unsafe
```



```
PS C:\Users\Administrator> .\godditest-windows-amd64.exe -username=juser -password="Fall2017!" -domain="demo.local" -dc=
"10.2.9.220" -unsafe_
```



Name	Date modified	Type	Size
Domain_Account_Policy.csv	4/16/2018 2:37 PM	CSV File	1 KB
Domain_Computers_All.csv	4/16/2018 2:37 PM	CSV File	2 KB
Domain_Controllers.csv	4/16/2018 2:37 PM	CSV File	1 KB
Domain_FSMO_Roles.csv	4/16/2018 2:37 PM	CSV File	1 KB
Domain_GPOs.csv	4/16/2018 2:37 PM	CSV File	2 KB
Domain_Groups_All.csv	4/16/2018 2:37 PM	CSV File	9 KB
Domain_MachineAccount_Old_Password...	4/16/2018 2:37 PM	CSV File	2 KB
Domain_OUs.csv	4/16/2018 2:37 PM	CSV File	1 KB
Domain_Passwords_GPP.csv	4/16/2018 2:37 PM	CSV File	3 KB
Domain_Passwords_LAPS.csv	4/16/2018 2:37 PM	CSV File	1 KB

POTENTIAL_SENSITIVE_DATA_FOUND.csv - Notepad
File Edit Format View Help

```
sAMAccountName,sAMAccountType,userPrincipalName,displayName,givenName,description,adminCount,homeDirectory,memberOf
juser,805306368,juser@demo.local,joe user,joe,password is falltest,,,juser,805306368,juser@demo.local,joe
user,joe,password is falltest,,,
```

Roadmap

In the future, I would love to see if I can get this tool to operate closer to the ADSI model. Being able to run the tool in the user's current context would be preferable from a testing standpoint. I would also like to improve how GPP passwords are gathered. Network shares to the target DC are mapped and mounted with the `net use` and `mount` commands. While GPP cpassword searching works with these commands, I have not gotten the chance to add robust error handling for corner cases when dealing with underlying OS errors.

GitHub Repo

Check out the goddi GitHub repo for install and usage instructions. I'll be updating the features list and roadmap there. Comments and commits are welcome! I'm not a Go expert, so I would appreciate any constructive feedback.