Product ⌄   Solutions ⌄   Resources ⌄   Open Source ⌄   Enterprise ⌄   Pricing   Sign in   Sign up

FalconForceTeam / **SOAPHound**   Public

🔔 Notifications   Fork 64   ☆ Star 645

`<>` Code   ⊙ Issues 4   Pull requests 1   ⊙ Actions   Security   Insights

main ⌄

Go to file   `<>` Code ⌄

0xFF-NK Update Program.cs •••	818a0b5 · 10 months ago	🕑 13 Commits
📁 ADWS	Update ADWSConnector.cs	10 months ago
📁 Enums	Initial commit	10 months ago
📁 OutputTypes	Initial commit	10 months ago
📁 Processors	Merge pull request #2 from eltociear/p...	10 months ago
📁 Properties	Initial commit	10 months ago
📄 .gitignore	Initial commit	10 months ago
📄 ADWSUtils.cs	Initial commit	10 months ago
📄 FodyWeavers.xml	Initial commit	10 months ago
📄 LICENSE	Initial commit	10 months ago
📄 Options.cs	Initial commit	10 months ago
📄 Program.cs	Update Program.cs	10 months ago
📄 README.md	Update README.md	10 months ago
📄 SOAPHound.csproj	Initial commit	10 months ago
📄 SOAPHound.sln	Initial commit	10 months ago
📄 hDNSRecord.cs	Initial commit	10 months ago
📄 packages.config	Initial commit	10 months ago
📄 soaphound-logo.png	Add files via upload	10 months ago

📖 README   ⚖ GPL-3.0 license   ☰

Description

About

SOAPHound is a custom-developed .NET data collector tool which can be used to enumerate Active Directory environments via the Active Directory Web Services (ADWS) protocol.

📖 Readme
⚖ GPL-3.0 license
〰 Activity
▤ Custom properties
☆ 645 stars
👁 11 watching
ᛉ 64 forks

Report repository

Releases

No releases published

Packages

No packages published

Contributors 4

👤 0xFF-NK
👤 olafhartong Olaf Hartong
👤 gijsh
👤 eltociear Ikko Eltociear Ashimine

Languages

● C# 100.0%

SOAPHound is a .NET data collector tool, which collects Active Directory data via the Active Directory Web Services (ADWS) protocol.

SOAPHound is an alternative to a number of open source security tools which are commonly used to extract Active Directory data via LDAP protocol. SOAPHound is able to extract the same information without directly communicating to the LDAP server. Instead, LDAP queries are wrapped within a series of SOAP messages, which are sent to the ADWS server using NET TCP Binding communication channel. Following, ADWS server unwraps the LDAP queries and forwards them to the LDAP server running on the same Domain Controller. As a result, LDAP traffic is not sent via the wire and therefore is not easily detected by common monitoring tools.

Note that this is a proof of concept tool and is not intended for production use. The tool is provided as is, without warranty of any kind.

For additional details on the SOAPHound tool, please refer to the following blog post: [SOAPHound — tool to collect Active Directory data via ADWS](https://github.com/FalconForceTeam/SOAPHound).

Usage

The `--help` command line argument can be used to display the following usage information:

```
SOAPHound
Copyright (c) 2024 FalconForce

Connection and authentication options:
  --user                  Username to use for ADWS Connection. Form
  --password              Password to use for ADWS Connection
  --domain                Specify domain for enumeration
  --dc                    Domain Controller to connect to

Supported collection methods:
  --buildcache            (Default: false) Only build cache and not
  --dnsdump               (Default: false) Dump AD Integrated DNS d
  --certdump              (Default: false) Dump AD Certificate Serv
  --bhdump                (Default: false) Dump BH data

Output options:
  -o, --outputdirectory   Folder to output files to (full path need
  -c, --cachefilename     Filename for the cache file (full path ne

Splitting options:
  -a, --autosplit         (Default: false) Enable AutoSplit mode: a
  -t, --threshold         (Default: 0) AutoSplit mode: Define split

Miscellaneous options:
  --nolaps                (Default: false) Do not request LAPS rela
  --showstats             Show stats of local cache file
  --logfile               Create log file
  --help                  Display this help screen.
```

Connection and authentication options

Authentication

SOAPHound supports the following authentication methods:

- Using the existing authentication token of the current user. This is the default option if no username and password are supplied.
- Supplying a username and password on the command line.

Domain Connection Information

When SOAPHound runs in a domain-joined machine, it will automatically attempt to connect to the Domain Controller of the domain the machine is joined to. This can be overridden by supplying the `--dc` and `--domain` command line arguments.

Supported collection methods

One of the following collection methods must be specified:

- `--buildcache` : Only build cache and not perform further actions
- `--bhdump` : Dump BloodHound data
- `--certdump` : Dump AD Certificate Services (ADCS) data
- `--dnsdump` : Dump AD Integrated DNS data

Building the cache

SOAPHound is able to generate a cache file that contains basic information about all domain objects, such as Security Identifier (SID), Distinguished Name (DN) and ObjectClass. This cache file is required for BloodHound related data collection (i.e. the `--bhdump` and `--certdump` collection methods), since it is used when crafting the trust relationships between objects via the relevant Access Control Entries (ACEs).

An example command to build the cache file is:

```
SOAPHound.exe --buildcache -c c:\temp\cache.txt
```

This will generate a cache file in the `c:\temp` folder. The cache file is a JSON formatted mapping of basic information about all domain objects. To view some statistics about the cache file (i.e. number of domain objects starting with each letter), you can use the `--showstats` command line argument:

```
SOAPHound.exe --showstats -c c:\temp\cache.txt
```

Collecting BloodHound Data

After the cache file has been generated, you can use the `--bhdump` collection method to collect data from the domain that can be imported into BloodHound.

An example command to collect BloodHound data is (note that this references the cache file generated in the previous step):

```
SOAPHound.exe -c c:\temp\cache.txt --bhdump -o c:\temp\bloodhound-ou·
```

If the targeted domain does not use LAPS, you can use the `--nolaps` command line argument to skip the LAPS related data collection.

This command will generate the `c:\temp\bloodhound-output` folder and produce a number of JSON files that can be imported into BloodHound. The JSON files contain the collected Users, Groups, Computers, Domains, GPOs and Containers, including their relationships. SOAPHound is compatible with Bloodhound version 4.

Dealing with large domains

If you are dealing with a large domain, you may run into issues with the amount of data that can be retrieved in a single request. To deal with this, SOAPHound supports the `--autosplit` and `--threshold` command line arguments.

The `--autosplit` command line argument enables the AutoSplit mode, which will automatically split object retrieval on two depth levels based on a defined threshold. The `--threshold` command line argument defines the split threshold based on the number of objects per starting letter.

An example command to collect BloodHound data in AutoSplit mode is:

```
SOAPHound.exe -c c:\temp\cache.txt --bhdump -o c:\temp\bloodhound-ou
```

This will generate the output in batches of a maximum of 1000 objects per starting letter. If there are more than 1000 objects for a single starting letter, SOAPHound will use two depth levels to retrieve the objects. This will result in larger number of queries, each one returning a maximum of 1000 objects.

For example if there are 2000 objects starting with the letter `a`, SOAPHound will retrieve objects starting with `aa`, `ab`, `ac`, etc., each in a separate query to avoid timeouts.

Collecting ADCS Data

After the cache file has been generated, you can use the `--certdump` collection method to collect ADCS data from the domain that can be imported into BloodHound. This collection method does not support the `--autosplit` and `--threshold` command line arguments.

An example command to collect ADCS data is (note that this references the cache file generated in previous step):

```
SOAPHound.exe -c c:\temp\cache.txt --certdump -o c:\temp\bloodhound-c
```

This command will generate the `c:\temp\bloodhound-output` folder and produce two JSON files that can be imported into BloodHound, containing information about the Certificate Authorities (CA) and Certificate Templates. SOAPHound is compatible with Bloodhound version 4 and ADCS data are classified as GPO objects in Bloodhound.

Collecting AD Integrated DNS Data

Apart from BloodHound data, SOAPHound can also be used to collect AD Integrated DNS data. This does not require a cache file and does not support the `--autosplit` and `--threshold` command line arguments.

An example command to collect AD Integrated DNS data is:

```
SOAPHound.exe --dnsdump -o c:\temp\dns-output
```

This command will generate a file namely DNS.txt in the `c:\temp\dns-output` folder that contains a dump of all the AD Integrated DNS data.

Acknowledgements

This tool is based on the work of the following open source projects:

- SharpHound
- StandIn
- Certify

Another big thanks to PingCastle for their reference implementation of the ADWS protocol. While we do not use their code directly, it was a great help in understanding the protocol and realizing the potential of the ADWS protocol.