Empire 1.5

Published March 31, 2016 by harmj0y

Three months have elapsed since the Empire 1.4 release, and we have some awesome new features for our next release! The notes for Empire 1.5 are below, but a quick warning- this release modifies part of the backend database schema, so **do not apply this update if you have existing agents on your Empire server**. You will need to run ./setup/reset.sh to reinitialize the database, and will likely need to rerun setup.sh or **pip install flask** to install the Flask dependencies necessary for the RESTful API.

New Modules

- The core version of PowerView was updated with the newest version from PowerSploit's dev branch. With that, several new **situational_awareness/network/powerview/*** modules were created:
  - **get_dfs_share** integrates some of meatballs' recent PowerView additions all fault-tolerant distributed file systems (DFS) for a given domain. This enumeration is also implicitly called by **user_hunter** when using the 'Stealth' option.
  - **get_domain_policy** will enumerate the default domain policy (things like the default KerberosPolicy). If you set **Source** to be 'DC' instead of 'Domain', the policy for the default domain controller will be enumerated instead (things like PrivilegeRights).
  - **get_fileserver** wraps Get-NetFileServer returns a list of all file servers extracted from user homedirectory, scriptpath, and profilepath fields. This is a nice compliment to **get_dfs_share**.

  - **get_gpo_computer** will take a GPO GUID and returns the computers the GPO is applied to. This can be really useful for things like figuring out what computers a discovered GPP password applies to.
  - **get_rdp_session** enumerates the remote (or local) RDP sessions on a remote machine that you have administrative access to. This is simliar to qwinsta, but also pulls in the originating IP of the connection as well. I have a post that describes this topic and the API calls in more depth here.
  - **get_site** returns a list of all current sites in the current (or specified) domain. This functionality was now also updated for **find_gpo_location**.
  - **get_subnet** enumerates all subnets in the current (or specified) domain. If you specify a **SiteName**, it will returns the subnets for the specified site.
  - **situational_awareness/host/get_proxy** will pull proxy server serrings and WPAD contents for the current user. You can also set a **ComputerName** to enumerate these settings on a remote machine.
  - **management/get_domain_sid** pulls the SID for the current of specified domain. This can be useful when constructing complex golden tickets.
  - **situational_awareness/host/get_pathacl** will enumerate the ACL for a given file path.
- **lateral_movement/new_gpo_immediate_task** implements what I spoke about in this post. The module lets you push out an 'immediate' scheduled task to a GPO that you can edit, allowing for code execution on systems where the GPO is applied.
- **privesc/getsystem** will execute getsystem type functionality. One note here- PowerShell 2.0 by default launches in a multi-threaded apartment. This prevents the current thread from sometimes doing correct impersonation, causing some methods like getsystem to fail. Empire's launchers are now spawned with the -sta argument (for single-threaded apartment) by default in order to try to resolve some of these issues.
- **privesc/mcafee_sitelist** implements PowerSploit's new Get-SiteListPassword function that retrieves the plaintext passwords for found McAfee's SiteList.xml files. This is heavily based on Jerome Nokin's excellent work on the subject.
- **trollsploit/rick_astley** is an audio beeping rickroll written by @SadProcessor.
- Inveigh was updated by the awesome @kevin_robertson! The Inveigh modules now include **collection/inveigh** for hash collection, **lateral_movement/inveigh_relay** for hash relay/lateral

movement, **collection/inveigh_bruteforce** to perform NBNS spoofing across subnets, and the **privesc/tater** implementation of the 'Hot Potato' exploit from @breenmachine and @foxglovesec.

Other Updates

As always, tons of bug fixes. Additionally, we implemented 'stager retries'. If you set the 'StagerRetries' option for any stager module, the basic staging code should retry for a connection the specified number of times.

We also have some more verbose debugging. Running "`./empire --debug`" will still output debug output to empire.debug, and now "`./empire --debug 2`" will up the verbosity level and display all debug output to the console as well as the log.

We've finally implemented a request from last year to support the loading of Empire modules from external directories, for the purposes of testing and private development. On the main menu, the '**load /path/to/folder**' command will now load modules from en external directory, usable with '**usemodule external/path/to/folder/module**'. Note that these modules do not persist across server restarts, but you can reload all modules in the folder with '**reload /path/to/folder**'.

GitHub Issue and Pull Request templates! We now have templates for contributing as well as issues. This helps remind contributors to include the information we need to help resolve your issue or approve your pull request.

Also, the epoch issue, oh the epoch issue. Empire has some attempted anti-replay functionality built in, where the client/sever sync up their system times on agent check in and each tasking/response packet contains a 32-bit counter field based on the epoch. The idea was to provide a +/- 10 minute sliding window where packets for packets to be validated. However, some users (particularly those operating across time zones) have reported issues with the client/server syncing their clocks, causing big problems in certain cases with agent taskings and results. This 'sliding window' has now been expanded to +/- 12 hours, still providing a bit of anti-replay resistance for historically captured traffic, but hopefully wide enough to fix the issue for anyone who's encountered it. If anyone continues to have epoch issues, please let us know.

Empire's RESTful API

Anyone following on twitter may have noticed a few tweets that I posted after the Troopers conference. @antisnatchor from the BeEF project **strongly** suggested that we implement a RESTful API for Empire, noting that the community started to do some awesome things with BeEF once the API was in place. A bit over a week later (plus some coffee, as well as a few beers) we have a prototype RESTful API implemented for Empire. Shoutout to BeEF as well for their excellent documentation on how their API works- it served as a great starting point. And also, I want to give a huge huge thank you to Carlos Perez for giving me a TON amount of feedback on the API right after seeing that tweet, and greatly guiding the API towards a better design.

Empire's RESTful API is implemented with Flask, and is based on Miguel Grinberg's excellent series of articles dealing with the subject. In the next year, we actually intend on moving all of Empire's http[s] server handling to Flask to avoid some of the concurrency issues we've heard occur when a high number of agents are registered with an Empire server.

In the next 1-2 weeks I'll have an in-depth blog post detailing the API model and what people can do with it. Until then, check out the comments for a brief description, and this gist for a brief overview of how to interact with the API. If you want to launch the API, you have two options: start a full `./empire` instance, and then run `./empire --rest --password X` from the same folder, or run `./empire --headless --password X` to kick everything off at once. Check **-h** for additional options, or stay tuned next week for a detailed writeup.

We look forward to what all of you can do with this new Empire integration!

Read more posts about Empire

Empire

Previous Post
Abusing GPO Permissions

Next Post
Empire’s RESTful API

Be First to Comment

Leave a Reply

Your email address will not be published. Required fields are marked *

Comment

Name*

Jane Doe

Email*

name@email.com

Website

http://google.com

□

Save my name, email, and website in this browser for the next time I comment.

Post Comment

□ Notify me of follow-up comments by email.

□ Notify me of new posts by email.

Cele Theme by Compete Themes.