# Contents

1 Cowrie
  1.1 Welcome to the Cowrie GitHub repository .......................... 1
  1.2 What is Cowrie .................................................. 1
  1.3 Documentation .................................................. 1
  1.4 Slack ............................................................ 1
  1.5 Features .......................................................... 1
  1.6 Docker ............................................................ 2
  1.7 Requirements .................................................... 2
  1.8 Files of interest: .................................................. 2
  1.9 I have some questions! ............................................. 3
  1.10 Contributors ..................................................... 3

2 Frequently asked questions ............................................. 5
  2.1 Why can’t I start cowrie on port 22? ............................. 5
  2.2 Why do I get logged into my own system when accessing cowrie on port 22? ............................. 5
  2.3 What I am getting permission denied when running cowrie on port 22? ............................. 5
  2.4 Do I need to copy all the content of cowrie.cfg.dist to cowrie.cfg? ............................. 5
  2.5 Why certain commands aren’t implemented? ............................. 6

3 LICENSE .......................... 7

4 Installing Cowrie in seven steps. .......................... 9
  4.1 Step 1: Install dependencies .................................... 9
  4.2 Step 2: Create a user account ................................... 10
  4.3 Step 3: Checkout the code ....................................... 10
  4.4 Step 4: Setup Virtual Environment ................................ 10
  4.5 Step 5: Install configuration file .............................. 11
  4.6 Step 6: Starting Cowrie ....................................... 11
  4.7 Step 7: Listening on port 22 (OPTIONAL) ..................... 11
  4.8 Running using Supervisord (OPTIONAL) ......................... 12
  4.9 Configure Additional Output Plugins (OPTIONAL) ................. 13

5 Troubleshooting ............................................. 15

6 Updating Cowrie ............................................. 17

7 Modifying Cowrie ............................................. 19
<table>
<thead>
<tr>
<th>Chapter</th>
<th>Title</th>
<th>Page</th>
</tr>
</thead>
<tbody>
<tr>
<td>8</td>
<td>Release 1.6.0</td>
<td>21</td>
</tr>
<tr>
<td>9</td>
<td>Release 1.5.3</td>
<td>23</td>
</tr>
<tr>
<td>10</td>
<td>Release 1.5.2</td>
<td>25</td>
</tr>
<tr>
<td>11</td>
<td>Release 1.5.1</td>
<td>27</td>
</tr>
<tr>
<td>12</td>
<td>Contributing Guidelines</td>
<td>31</td>
</tr>
<tr>
<td>13</td>
<td>Reporting Bugs/Feature Requests</td>
<td>33</td>
</tr>
<tr>
<td>14</td>
<td>Contributing via Pull Requests</td>
<td>35</td>
</tr>
<tr>
<td>15</td>
<td>Finding contributions to work on</td>
<td>37</td>
</tr>
<tr>
<td>16</td>
<td>Licensing</td>
<td>39</td>
</tr>
<tr>
<td>17</td>
<td>How to process Cowrie output in an ELK stack</td>
<td>41</td>
</tr>
<tr>
<td>17.1</td>
<td>Prerequisites</td>
<td>41</td>
</tr>
<tr>
<td>17.2</td>
<td>Installation</td>
<td>41</td>
</tr>
<tr>
<td>17.3</td>
<td>ElasticSearch Configuration</td>
<td>42</td>
</tr>
<tr>
<td>17.4</td>
<td>Kibana Configuration</td>
<td>42</td>
</tr>
<tr>
<td>17.5</td>
<td>Logstash Configuration</td>
<td>42</td>
</tr>
<tr>
<td>17.6</td>
<td>Distributed setup of sensors or multiple sensors on the same host</td>
<td>43</td>
</tr>
<tr>
<td>17.7</td>
<td>Tuning ELK stack</td>
<td>44</td>
</tr>
<tr>
<td>18</td>
<td>How to process Cowrie output into Graylog</td>
<td>45</td>
</tr>
<tr>
<td>18.1</td>
<td>Prerequisites</td>
<td>45</td>
</tr>
<tr>
<td>18.2</td>
<td>Cowrie Configuration</td>
<td>45</td>
</tr>
<tr>
<td>18.3</td>
<td>Graylog Configuration</td>
<td>45</td>
</tr>
<tr>
<td>18.4</td>
<td>Syslog Configuration</td>
<td>46</td>
</tr>
<tr>
<td>19</td>
<td>How to process Cowrie output in kippo-graph</td>
<td>47</td>
</tr>
<tr>
<td>19.1</td>
<td>Prerequisites</td>
<td>47</td>
</tr>
<tr>
<td>19.2</td>
<td>Installation</td>
<td>47</td>
</tr>
<tr>
<td>19.3</td>
<td>MySQL configuration</td>
<td>47</td>
</tr>
<tr>
<td>19.4</td>
<td>Cowrie configuration</td>
<td>48</td>
</tr>
<tr>
<td>19.5</td>
<td>kippo-graph Configuration</td>
<td>48</td>
</tr>
<tr>
<td>19.6</td>
<td>Apache2 configuration (optional)</td>
<td>49</td>
</tr>
<tr>
<td>20</td>
<td>How to process Cowrie output with Splunk</td>
<td>51</td>
</tr>
<tr>
<td>20.1</td>
<td>Splunk Output Module</td>
<td>51</td>
</tr>
<tr>
<td>20.2</td>
<td>File Based</td>
<td>51</td>
</tr>
<tr>
<td>20.3</td>
<td>Reporting</td>
<td>51</td>
</tr>
<tr>
<td>21</td>
<td>How to Send Cowrie Output to a MySQL Database</td>
<td>53</td>
</tr>
<tr>
<td>21.1</td>
<td>Prerequisites</td>
<td>53</td>
</tr>
<tr>
<td>21.2</td>
<td>Installation</td>
<td>53</td>
</tr>
<tr>
<td>21.3</td>
<td>MySQL Configuration</td>
<td>53</td>
</tr>
<tr>
<td>21.4</td>
<td>Cowrie Configuration</td>
<td>54</td>
</tr>
<tr>
<td>22</td>
<td>Using TCP tunneling with Squid</td>
<td>57</td>
</tr>
<tr>
<td>22.1</td>
<td>Prerequisites</td>
<td>57</td>
</tr>
<tr>
<td>22.2</td>
<td>Installation</td>
<td>57</td>
</tr>
<tr>
<td>22.3</td>
<td>Squid Configuration</td>
<td>57</td>
</tr>
</tbody>
</table>
1.1 Welcome to the Cowrie GitHub repository

This is the official repository for the Cowrie SSH and Telnet Honeypot effort.

1.2 What is Cowrie

Cowrie is a medium interaction SSH and Telnet honeypot designed to log brute force attacks and the shell interaction performed by the attacker.

Cowrie is developed by Michel Oosterhof.

1.3 Documentation

The Documentation can be found here.

1.4 Slack

You can join the Cowrie community at the following Slack workspace.

1.5 Features

Some interesting features:
Fake filesystem with the ability to add/remove files. A full fake filesystem resembling a Debian 5.0 installation is included

- Possibility of adding fake file contents so the attacker can `cat` files such as `/etc/passwd`. Only minimal file contents are included
- Session logs are stored in an UML Compatible format for easy replay with original timings with the `bin/playlog` utility.
- Cowrie saves files downloaded with `wget/curl` or uploaded with SFTP and scp for later inspection log

Additional functionality over standard kippo:

- SFTP and SCP support for file upload
- Support for SSH exec commands
- Logging of direct-tcp connection attempts (ssh proxying)
- Forward SMTP connections to SMTP Honeypot (e.g. mailoney)
- Logging in JSON format for easy processing in log management solutions
- Many, many additional commands

### 1.6 Docker

Docker versions are available.

- To get started quickly and give Cowrie a try, run:
  
  ```
  docker run -p 2222:2222 cowrie/cowrie
  ssh -p 2222 root@localhost
  ```

- On Docker Hub: https://hub.docker.com/r/cowrie/cowrie
- Or get the Dockerfile directly at https://github.com/cowrie/docker-cowrie

### 1.7 Requirements

Software required:

- Python 3.5+ (Python 2.7 supported for now but we recommend to upgrade)
- python-virtualenv

For Python dependencies, see requirements.txt.

### 1.8 Files of interest:

- `cowrie.cfg` - Cowrie’s configuration file. Default values can be found in etc/cowrie.cfg.dist.
- `share/cowrie/fs.pickle` - fake filesystem
- `etc/userdb.txt` - credentials allowed or disallowed to access the honeypot
- `honeyfs/` - file contents for the fake filesystem - feel free to copy a real system here or use `bin/fsctl`
- `honeyfs/etc/issue.net` - pre-login banner
• honeyfs/etc/motd - post-login banner
• var/log/cowrie/cowrie.json - transaction output in JSON format
• var/log/cowrie/cowrie.log - log/debug output
• var/lib/cowrie/tty/ - session logs, replayable with the bin/playlog utility.
• var/lib/cowrie/downloads/ - files transferred from the attacker to the honeypot are stored here
• share/cowrie/txtcmds/ - file contents for simple fake commands
• bin/createfs - used to create the fake filesystem
• bin/playlog - utility to replay session logs

1.9 I have some questions!

Please visit the Slack workspace and join the #questions channel.

1.10 Contributors

Many people have contributed to Cowrie over the years. Special thanks to:
• Upi Tamminen (desaster) for all his work developing Kippo on which Cowrie was based
• Dave Germiquet (davegermiquet) for TFTP support, unit tests, new process handling
• Olivier Bilodeau (obilodeau) for Telnet support
• Ivan Korolev (fe7ch) for many improvements over the years.
• Florian Pelgrim (craneworks) for his work on code cleanup and Docker.
• And many many others.
2.1 Why can’t I start cowrie on port 22?

The possible answer for that is you might already have a service (possibly SSH) running on that port so setting up Cowrie on that port will cause a problem. Try changing the port in listen_endpoints present in config file (cowrie.cfg.dist/cowrie.cfg).

2.2 Why do I get logged into my own system when accessing cowrie on port 22?

This is probably a similar problem as it was in the above question. This can also be fixed by changing the port in the config file.

2.3 What I am getting permission denied when running cowrie on port 22?

You need root privileges to run Cowrie on any port lower than 1024. This can be fixed by setting up Authbind.

2.4 Do I need to copy all the content of cowrie.cfg.dist to cowrie.cfg?

No, Cowrie can read only your local changes to cowrie.cfg and the remaining settings will automatically be read from cowrie.cfg.dist.
2.5 Why certain commands aren't implemented?

There are lots of UNIX command implemented in cowrie and that is because Cowrie is more focused to provide proxy support i.e use Cowrie to connect to an actual machine that is actual machine having support for all the UNIX functionalities.
Copyright (c) 2009-2019 Upi Tamminen, Michel Oosterhof
All rights reserved.
Redistribution and use in source and binary forms, with or without modification, are permitted provided that the following conditions are met:

- Redistributions of source code must retain the above copyright notice, this list of conditions and the following disclaimer.
- Redistributions in binary form must reproduce the above copyright notice, this list of conditions and the following disclaimer in the documentation and/or other materials provided with the distribution.
- The names of the author(s) may not be used to endorse or promote products derived from this software without specific prior written permission.

THIS SOFTWARE IS PROVIDED BY THE AUTHORS “AS IS” AND ANY EXPRESS OR IMPLIED WARRANTIES, INCLUDING, BUT NOT LIMITED TO, THE IMPLIED WARRANTIES OF MERCHANTABILITY AND FITNESS FOR A PARTICULAR PURPOSE ARE DISCLAIMED. IN NO EVENT SHALL THE AUTHORS BE LIABLE FOR ANY DIRECT, INDIRECT, INCIDENTAL, SPECIAL, EXEMPLARY, OR CONSEQUENTIAL DAMAGES (INCLUDING, BUT NOT LIMITED TO, PROCUREMENT OF SUBSTITUTE GOODS OR SERVICES; LOSS OF USE, DATA, OR PROFITS; OR BUSINESS INTERRUPTION) HOWEVER CAUSED AND ON ANY THEORY OF LIABILITY, WHETHER IN CONTRACT, STRICT LIABILITY, OR TORT (INCLUDING NEGLIGENCE OR OTHERWISE) ARISING IN ANY WAY OUT OF THE USE OF THIS SOFTWARE, EVEN IF ADVISED OF THE POSSIBILITY OF SUCH DAMAGE.
Installing Cowrie in seven steps.

4.1 Step 1: Install dependencies

First we install system-wide support for Python virtual environments and other dependencies. Actual Python packages are installed later.

On Debian based systems (last verified on Debian 9, 2017-07-25): For a Python3 based environment:

```
$ sudo apt-get install git python-virtualenv libssl-dev libffi-dev build-essential 
   -libpython3-dev python3-minimal authbind
```

Or for Python2:

```
$ sudo apt-get install git python-virtualenv libssl-dev libffi-dev build-essential 
   -libpython-dev python2.7-minimal authbind
```
4.2 Step 2: Create a user account

It's strongly recommended to run with a dedicated non-root user id:

```
$ sudo adduser --disabled-password cowrie
Adding user 'cowrie' ...
Adding new group 'cowrie' (1002) ...
Adding new user 'cowrie' (1002) with group 'cowrie' ...
Changing the user information for cowrie
Enter the new value, or press ENTER for the default
Full Name []:
Room Number []:
Work Phone []:
Home Phone []:
Other []:
Is the information correct? [Y/n]
$ sudo su - cowrie
```

4.3 Step 3: Checkout the code

Check out the code:

```
$ git clone http://github.com/cowrie/cowrie
Cloning into 'cowrie'... 
remote: Counting objects: 2965, done.
remote: Compressing objects: 100% (1025/1025), done.
remote: Total 2965 (delta 1908), reused 2962 (delta 1905), pack-reused 0
Receiving objects: 100% (2965/2965), 3.41 MiB | 2.57 MiB/s, done.
Resolving deltas: 100% (1908/1908), done.
Checking connectivity... done.
$ cd cowrie
```

4.4 ## Step 4: Setup Virtual Environment

Next you need to create your virtual environment:

```
$ pwd
/home/cowrie/cowrie
$ virtualenv --python=python3 cowrie-env
New python executable in ./cowrie/cowrie-env/bin/python
Installing setuptools, pip, wheel...done.
```

Alternatively, create a Python2 virtual environment:

```
$ virtualenv --python=python2 cowrie-env
New python executable in ./cowrie/cowrie-env/bin/python
Installing setuptools, pip, wheel...done.
```

Activate the virtual environment and install packages:
4.5 Step 5: Install configuration file

The configuration for Cowrie is stored in cowrie.cfg.dist and cowrie.cfg. Both files are read on startup, where entries from cowrie.cfg take precedence. The .dist file can be overwrittten by upgrades, cowrie.cfg will not be touched. To run with a standard configuration, there is no need to change anything. To enable telnet, for example, create cowrie.cfg and input only the following:

```
[telnet]
enabled = true
```

4.6 Step 6: Starting Cowrie

Start Cowrie with the cowrie command. You can add the cowrie/bin directory to your path if desired. An existing virtual environment is preserved if activated, otherwise Cowrie will attempt to load the environment called "cowrie-env":

```
$ bin/cowrie start
Activating virtualenv "cowrie-env"
Starting cowrie with extra arguments [] ...
```

4.7 Step 7: Listening on port 22 (OPTIONAL)

There are three methods to make Cowrie accessible on the default SSH port (22): `iptables`, `authbind` and `setcap`.

4.7.1 Iptables

Port redirection commands are system-wide and need to be executed as root. A firewall redirect can make your existing SSH server unreachable, remember to move the existing server to a different port number first.

The following firewall rule will forward incoming traffic on port 22 to port 2222 on Linux:

```
$ sudo iptables -t nat -A PREROUTING -p tcp --dport 22 -j REDIRECT --to-port 2222
```

Or for telnet:

```
$ sudo iptables -t nat -A PREROUTING -p tcp --dport 23 -j REDIRECT --to-port 2223
```

Note that you should test this rule only from another host; it doesn’t apply to loopback connections.

On MacOS run:

```
$ echo "rdr pass inet proto tcp from any to any port 22 -> 127.0.0.1 port 2222" | sudo pfctl -ef -
```

4.5. Step 5: Install configuration file
### 4.7.2 Authbind

Alternatively you can run authbind to listen as non-root on port 22 directly:

```bash
$ sudo apt-get install authbind
$ sudo touch /etc/authbind/byport/22
$ sudo chown cowrie:cowrie /etc/authbind/byport/22
$ sudo chmod 770 /etc/authbind/byport/22
```

Edit bin/cowrie and modify the AUTHBIND_ENABLED setting

Change the listening port to 22 in cowrie.cfg:

```
[ssh]
listen_endpoints = tcp:22:interface=0.0.0.0
```

Or for telnet:

```bash
$ apt-get install authbind
$ sudo touch /etc/authbind/byport/23
$ sudo chown cowrie:cowrie /etc/authbind/byport/23
$ sudo chmod 770 /etc/authbind/byport/23
```

Change the listening port to 23 in cowrie.cfg:

```
[telnet]
listen_endpoints = tcp:2223:interface=0.0.0.0
```

### 4.7.3 Setcap

Or use setcap to give permissions to Python to listen on ports<1024:

```bash
$ setcap cap_net_bind_service=+ep /usr/bin/python2.7
```

And change the listening ports in `cowrie.cfg` as above.

### 4.8 Running using Supervisord (OPTIONAL)

On Debian, put the below in `/etc/supervisor/conf.d/cowrie.conf`:

```
[program:cowrie]
command=/home/cowrie/cowrie/bin/cowrie start
directory=/home/cowrie/cowrie/
user=cowrie
autorestart=true
redirect_stderr=true
```

Update the bin/cowrie script, change:

```
DAEMONIZE=""
```

to:
4.9 Configure Additional Output Plugins (OPTIONAL)

Cowrie automatically outputs event data to text and JSON log files in `var/log/cowrie`. Additional output plugins can be configured to record the data other ways. Supported output plugins include:

- Cuckoo
- ELK (Elastic) Stack
- Graylog
- Kippo-Graph
- Splunk
- SQL (MySQL, SQLite3, RethinkDB)

See `~/cowrie/docs/[Output Plugin]/README.rst` for details.
Troubleshooting

If you see twisted: Unknown command: cowrie there are two possibilities. If there’s a Python stack trace, it probably means there’s a missing or broken dependency. If there’s no stack trace, double check that your PYTHONPATH is set to the source code directory.

Default file permissions

To make Cowrie logfiles public readable, change the --umask 0077 option in start.sh into --umask 0022
Updating is an easy process. First stop your honeypot. Then fetch updates from GitHub, and upgrade your Python dependencies:

```
bin/cowrie stop
git pull
pip install --upgrade -r requirements.txt
```

If you use output plugins like SQL, Splunk, or ELK, remember to also upgrade your dependencies for these too:

```
pip install --upgrade -r requirements-output.txt
```

And finally, start Cowrie back up after finishing all updates:

```
bin/cowrie start
```
Modifying Cowrie

The pre-login banner can be set by creating the file `honeyfs/etc/issue.net`. The post-login banner can be customized by editing `honeyfs/etc/motd`.

- 2019-06-20 Move `auth_none` and `auth_keyboard_interactive_enabled` to `[ssh]` config section
• 2019-03-31 New documentation theme
• 2019-03-23 GreyNoise output plugin (@mzfr)
• 2019-03-19 direct-tcp forwarding now written to databases (@gborges)
• 2019-03-19 Reverse DNS output plugin (@mzfr)
• 2019-03-17 Shell emulation pipe upgrade (@nunonovais)
• 2019-03-14 Shell emulation environment variables improved (@nunonovais)
• 2019-03-14 SSH crypto parameters now configurable in config file (@msharma)
• 2019-03-13 Disable keyboard-interactive authentication by default with option to enable
• 2019-03-13 Added *wc, crontab, chpasswd* command (@nunonovais)
• 2019-
• 2019-03-07 Output of *ssh -V* now configurable in cowrie.cfg with *ssh_version* setting
• 2019-03-07 Multiple timezone support in cowrie.cfg timezone directive. Default timezone is now UTC for both cowrie.log and cowrie.json
• 2019-03-12 Handle multiple password prompt. Option to enable or disable keyboard interactive prompt.
• 2019-01-27 Telnet NAWS negotiation removed to stop NMAP cowrie detection
• 2019-01-27 Various fixes for Python2/3 compatibility
• 2019-01-09 Documentation converted to ReStructuredText
• 2018-12-04 Fixes for VT output plugin to only submit new files
• 2018-11-19 Fix tftp exception and tftp test
• 2018-11-14 Remove *dblog* mechanism and *splunk* legacy output plugin.
• 2018-11-01 Add Python3 support for Splunk output plugin
• 2018-10-23 Improved free command
• 2018-10-20 Improved uname command
• 2018-10-16 Save VT results to JSON log
• 2018-10-13 Fixes VT uploads, tab completion on Python3, Hassh support, setuptools functional. userdb migration
• 2018-09-07 NOTE! data/userdb.txt has moved to etc/userdb.txt and a default config is no longer provided!
• 2018-08-25 Downloads and TTY logs have moved to the var/ directory
• 2018-08-11 SSH keys now stored in var/lib/cowrie
• 2018-07-21 source code has move to the src/ directory. Delete old directories twisted/cowrie with compiled code
• 2018-06-29 txtcmds have been moved to share/cowrie/txtcmds
• 2018-06-28 filesystem config entry has changed. please verify if you have custom entry or pickle file
• 2018-06-23 fingerprint log message now holds KEX attributes and a unique fingerprint for the client
• 2018-04-27 Output plugins now require the mandatory config entry ‘enabled’.
• 2018-02-06 cowrie.log now uses same rotation mechanism as cowrie.json. One file per day, rather than the default 1MB per file.
• 2017-12-13 Default umask for logs is now 0007. This means group members can access.
• 2017-10-24 Can store uploaded and downloaded artifacts to S3
• 2017-09-23 First proxy implementation for exec commands only
• 2017-07-03 Cuckoo v2 integration
• 2017-05-16 now combines config files: cowrie.cfg.dist and cowrie.cfg in this order
• 2017-05-09 start.sh and stop.sh have been replace by bin/cowrie start|stop
• 2017-04-27 New syntax “listen_endpoints” for configuring listening IP addresses/portnumbers
• 2017-03-15 SSH Forwarding/SFTP/keys/version config have been moved to [ssh]. Change your config file!
• 2017-02-12 Implemented toggle for SSH forwarding
• 2016-08-22 Merged Telnet support by @obilodeau!
• 2016-08-20 Update your libraries! ‘configparser’ now required: “pip install configparser”
• 2016-05-06 Load pickle once at startup for improved speed
• 2016-04-28 files in utils/ have been moved to bin/
• 2016-01-19 Support openssh style delayed compression
• 2016-01-13 Correct ‘.’ support and +s and +t bits in ls
• 2016-01-13 Full username/group in SFTP ls
• 2016-01-05 Basic VirusTotal support has been added
• 2016-01-04 No longer crash when client tries ecdsa
• 2015-12-28 Interact port (default 5123) only listens on loopback interface now (127.0.0.1)
• 2015-12-24 Redirect to file (>) now works for most commands and is logged in dl/ directory
• 2015-12-06 UID information is now retrieved from honeyfs/etc/passwd. If you added additional users you will need to add these to the passwd file as well
• 2015-12-04 New ‘free’ command with ‘-h’ and ‘-m’ options
• 2015-12-03 New ‘env’ command that prints environment variables
• 2015-02-02 Now use honeyfs/etc/passwd and group to get uid/gid info
• 2015-11-29 Size limit now enforced for SFTP uploads
• 2015-11-25 New ‘sudo’ command added
• 2015-11-19 Queued input during commands is now sent to shell to be executed when command is finished
• 2015-11-18 Added SANS DShield output (Thanks @UnrealAkama)
• 2015-11-17 Added ElasticSearch output (Thanks @UnrealAkama)
• 2015-11-17 Standard input is now saved with SHA256 checksum. Duplicate data is not saved
• 2015-11-12 New ‘busybox’ command added (Thanks @mak)
• 2015-09-26 keyboard-interactive is back as authentication method, after Twisted removed support initially
• 2015-07-30 Local syslog output module
• 2015-06-15 Cowrie now has a ‘-c’ startup switch to specify the configuration file
• 2015-06-15 Removed exec_enabled option. This feature is now always enabled
• 2015-06-03 Cowrie now uses twisted plugins and has gained the ‘-p’ commandline option
• 2015-06-01 Cowrie no longer search for config files in /etc and /etc/cowrie
• 2015-04-12 JSON output is now default via ‘output’ plugin mechanism. Rotates daily
• 2015-04-10 Fix for downloading files via SFTP
• 2015-03-31 Small tweaks on session close, closing session does not close ssh transport
• 2015-03-18 Merged ‘AuthRandom’ login class by Honigbij
• 2015-02-25 Internals for dblog/ modules changed completely. Now accepts structured logging arguments, and uses eventids instead of regex parsing
• 2015-02-20 Removed screen clear/reset on logout
• 2015-02-19 Configuration directives have changed! ssh_addr has become listen_addr and ssh_port has become listen_port. The old keywords are still accepted for backwards compatibility
• default behaviour is changed to disable the exit jail
• sftp support
• exec support

• stdin is saved as a file in dl/ when using exec commands to support commands like ‘cat >file; ./file’
• allow wget download over non-80 port
• simple JSON logging added
• accept log and deny publickey authentication
• add uname -r, -m flags
• add working sleep command
• enabled ssh diffie-hellman-group-exchange-sha1 algorithm
• add ‘bash -c’ support (no effect option)
• enable support for && multiple commands
• create uuid to uniquely identify each session
• log and deny direct-tcpip attempts
• add “chattr” command
• support emacs keybindings (c-a, c-b, c-f, c-p, c-n, c-e)
• add “sync” command
• accept, log and deny public key authentication
• add “uname -r” support
• logstash and kibana config files added, based on JSON log
• fix for honeypot detection (pre-auth differences with openssh)
• added verbose logging of client requested key exchange parameters (for client fingerprinting)
• fixes for behavior with non-existent files (cd /test, cat /test/nonexistent, etc)
• fix for ability to ping/ssh non-existent IP address
• always send ssh exit-status 0 on exec and shell
• ls output is now alphabetically sorted
• banner_file is deprecated. honeyfs/etc/issue.net is default
• add ‘dir’ alias for ‘ls’
• add ‘help’ bash builtin
• add ‘users’ aliased to ‘whoami’
• add ‘killall’ and ‘killall5’ aliased to nop
• add ‘poweroff’ ‘halt’ and ‘reboot’ aliases for shutdown
• add environment passing to commands
• added ‘which’, ‘netstat’ and ‘gcc’ from kippo-extra
• logging framework allows for keyword use
Thank you for your interest in contributing to our project. Whether it’s a bug report, new feature, correction, or additional documentation, we greatly value feedback and contributions from our community.

Please read through this document before submitting any issues or pull requests to ensure we have all the necessary information to effectively respond to your bug report or contribution.
We welcome you to use the GitHub issue tracker to report bugs or suggest features. When filing an issue, please check existing open, or recently closed, issues to make sure somebody else hasn’t already reported the issue. Please try to include as much information as you can. Details like these are incredibly useful:

- A reproducible test case or series of steps
- The version of our code being used
- Any modifications you’ve made relevant to the bug
- Anything unusual about your environment or deployment
Contributing via Pull Requests

Contributions via pull requests are much appreciated. Before sending us a pull request, please ensure that:

1. You are working against the latest source on the master branch.
2. You check existing open, and recently merged, pull requests to make sure someone else hasn’t addressed the problem already.
3. You open an issue to discuss any significant work - we would hate for your time to be wasted.

To send us a pull request, please:

1. Fork the repository.
2. Modify the source; please focus on the specific change you are contributing. If you also reformat all the code, it will be hard for us to focus on your change.
3. Ensure local tests pass.
4. Commit to your fork using clear commit messages.
5. Send us a pull request, answering any default questions in the pull request interface.
6. Pay attention to any automated CI failures reported in the pull request, and stay involved in the conversation.

GitHub provides additional document on forking a repository and creating a pull request.
Finding contributions to work on

Looking at the existing issues is a great way to find something to contribute on. As our projects, by default, use the default GitHub issue labels ((enhancement/bug/duplicate/help wanted/invalid/question/wontfix), looking at any help wanted issues is a great place to start.
See the LICENSE file for our project’s licensing. We will ask you confirm the licensing of your contribution.
CHAPTER 17

How to process Cowrie output in an ELK stack

(Note: work in progress, instructions are not verified)

17.1 Prerequisites

- Working Cowrie installation
- Cowrie JSON log file (enable database json in cowrie.cfg)
- Java 8

17.2 Installation

We’ll examine simple installation, when we install ELK stack on the same machine that used for cowrie.

Add Elastic’s repository and key:

```
wget -qO - https://packages.elastic.co/GPG-KEY-elasticsearch | sudo apt-key add -
echo "deb https://artifacts.elastic.co/packages/5.x/apt stable main" | sudo tee -a /etc/apt/sources.list.d/elastic-5.x.list
apt-get update
```

Install logstash, elasticsearch and kibana:

```
sudo apt-get install elasticsearch logstash kibana
```

Set them to autostart:

```
sudo update-rc.d elasticsearch defaults 95 10
dsudo update-rc.d logstash defaults 95 10
```
17.3 ElasticSearch Configuration

TBD

17.4 Kibana Configuration

Make a folder for logs:

```
sudo mkdir /var/log/kibana
sudo chown kibana:kibana /var/log/kibana
```

Change the following parameters in `/etc/kibana/kibana.yml` to reflect your server setup:

```
"server.host" - set it to "localhost" if you use nginx for basic authentication or external interface if you use XPack (see below)
"server.name" - name of the server
"elasticsearch.url" - address of the elasticsearch
"elasticsearch.username", "elasticsearch.password" - needed only if you use XPack (see below)
"logging.dest" - set path to logs (/var/log/kibana/kibana.log)
```

Make sure the file `/etc/kibana/kibana.yml` contains a line like:

```
tilemap.url: https://tiles.elastic.co/v2/default/{z}/{x}/{y}.png?elastic_tile_service=tos=agree&my_app_name=kibana
```

or your map visualizations won’t have any background. When this file is created during the installation of Kibana, it does _not_ contain such a line, not even in commented out form.

17.5 Logstash Configuration

Download GeoIP data:

```
wget http://geolite.maxmind.com/download/geoip/database/GeoLite2-City.mmdb.gz
```

Place these somewhere in your filesystem and make sure that “logstash” user can read it:

```
sudo mkdir -p /var/opt/logstash/vendor/geoip/
sudo mv GeoLite2-City.mmdb /var/opt/logstash/vendor/geoip
```

Configure logstash:

```
sudo cp logstash-cowrie.conf /etc/logstash/conf.d
```

Make sure the configuration file is correct. Check the input section (path), filter (geoip databases) and output (elasticsearch hostname):

```
sudo service logstash restart
```

By default the logstash is creating debug logs in /tmp.
To test whether logstash is working correctly, check the file in /tmp:
To test whether data is loaded into ElasticSearch, run the following query:

```
curl 'http://<hostname>:9200/_search?q=cowrie&size=5'
```

(Replace `<hostname>` with the name or IP address of the machine on which ElasticSearch is running, e.g., `localhost`.)

If this gives output, your data is correctly loaded into ElasticSearch.

When you successfully configured logstash, remove “file” and “stdout” blocks from output section of logstash configuration.

## 17.6 Distributed setup of sensors or multiple sensors on the same host

If you have multiple sensors, you will need to setup up FileBeat to feed logstash with logs from all sensors.

On the logstash server:

Change “input” section of the logstash to the following:

```yaml
input {
  beats {
    port => 5044
  }
}
```

On the sensor servers:

Install filebeat:

```
wget -qO - https://packages.elastic.co/GPG-KEY-elasticsearch | sudo apt-key add -
echo "deb https://artifacts.elastic.co/packages/5.x/apt stable main" | sudo tee -a /etc/apt/sources.list.d/elastic-5.x.list
sudo apt-get update
sudo apt-get install filebeat
```

Enable autorun for it:

```
sudo update-rc.d filebeat defaults 95 10
```

Configure filebeat:

```
sudo cp filebeat-cowrie.conf /etc/filebeat/filebeat.yml
```

Check the following parameters:

```
paths - path to cowrie's json logs
logstash - check ip of the logstash host
```

Start filebeat:

```
sudo service filebeat start
```
17.7 Tuning ELK stack

Refer to elastic’s documentation about proper configuration of the system for the best elasticsearch’s performance.

You may avoid installing nginx for restricting access to kibana by installing official elastic’s plugin called “X-Pack” (https://www.elastic.co/products/stack)
18.1 Prerequisites

- Working Cowrie installation
- Working Graylog installation

18.2 Cowrie Configuration

Open the Cowrie configuration file and uncomment these 3 lines:

```plaintext
[output_localsyslog]
facility = USER
format = text
```

Restart Cowrie

18.3 Graylog Configuration

Open the Graylog web interface and click on the System drop-down in the top menu. From the drop-down menu select Inputs. Select Syslog UDP from the drop-down menu and click the Launch new input button. In the modal dialog enter the following information:

```plaintext
**Title:** Cowrie  
**Port:** 8514  
**Bind address:** 127.0.0.1
```

Then click Launch.
18.4 Syslog Configuration

Create a rsyslog configuration file in /etc/rsyslog.d:

```
$ sudo nano /etc/rsyslog.d/85-graylog.conf
```

Add the following lines to the file:

```
$template GRAYLOGRFC5424,"<%pri%>%protocol-version% %timestamp:::date-rfc3339% →%HOSTNAME% %app-name% %procid% %msg%\n"
*.@127.0.0.1:8514;GRAYLOGRFC5424
```

Save and quit.

Restart rsyslog:

```
$ sudo service rsyslog restart
```
How to process Cowrie output in kippo-graph

(Note: work in progress, instructions are not verified) Tested on Debian 9.

19.1 Prerequisites

- Working Cowrie installation
- LAMP stack (Linux, Apache, MySQL, PHP)

19.2 Installation

This covers a simple installation, with kippo-graph and Cowrie on the same server. Please see here for installation: https://github.com/ikoniaris/kippo-graph

19.3 MySQL configuration

Configuring Cowrie requires setting up the SQL tables and then telling Cowrie to use them.

To install the tables and create the Cowrie user account enter the following commands:

```bash
mysql -u root -p
cREATE DATABASE cowrie;
GRANT ALL ON cowrie.* TO 'cowrie'@'localhost' IDENTIFIED BY 'PASSWORD HERE';
FLUSH PRIVILEGES;
exit
```

Next create the database schema:
disable MySQL strict mode:

```bash
vi /etc/mysql/conf.d/disable_strict_mode.cnf

[mysqld]
sql_mode=IGNORE_SPACE,NO_ZERO_IN_DATE,NO_ZERO_DATE,ERROR_FOR_DIVISION_BY_ZERO,NO_AUTO_CREATE_USER,NO_ENGINE_SUBSTITUTION
```

### 19.4 Cowrie configuration

Edit cowrie.cfg:

```bash
vi /opt/cowrie/cowrie.cfg
```

Activate output to mysql:

```ini
[output_mysql]
host = localhost
database = cowrie
username = cowrie
password = PASSWORD HERE
port = 3306
debug = false
```

Set read access to tty-files for group www-data (group maybe differ on other distributions):

```bash
sudo apt-get install acl
sudo setfacl -Rm g:www-data:rx /opt/cowrie/var/lib/cowrie/tty/
```

### 19.5 kippo-graph Configuration

Edit config file:

```bash
vi /var/www/html/kippo-graph/config.php
```

Change db settings:

```php
define('DB_HOST', 'localhost');
define('DB_USER', 'cowrie');
define('DB_PASS', 'PASSWORD HERE');
define('DB_NAME', 'cowrie');
define('DB_PORT', '3306');
```
19.6 Apache2 configuration (optional)

To secure the installation

Create password database:

```
cd /etc/apache2/
htpasswd -c /etc/apache2/cowrie.passwd <username>
htpasswd /etc/apache2/cowrie.passwd <username> (second user)
```

```
vi /etc/apache2/sites-enabled/000-default.conf
```

Between the `<VirtualHost>` </VirtualHost> tags, add:

```
<Location />
  AuthBasicAuthoritative On
  AllowOverride AuthConfig

  AuthType Basic
  AuthName "cowrie honeypot"
  AuthUserFile /etc/apache2/cowrie.passwd
  Require valid-user
</Location>
```
How to process Cowrie output with Splunk

20.1 Splunk Output Module

- In Splunk, enable the HTTP Event Collector (go to Settings->Add Data)
- Do not enable Indexer Acknowledgment
- Copy the authorization token for later use
- Modify cowrie.cfg to enable the [splunk] section
- Add URL to HTTP Event Collector and add the authorization token
- Optionally enable sourcetype, source, host and index settings

20.2 File Based

- Collect cowrie.json output file using Splunk

20.3 Reporting

Please see: https://github.com/aplura/Tango
How to Send Cowrie Output to a MySQL Database

21.1 Prerequisites

- Working Cowrie installation
- MySQL Server installation

21.2 Installation

Run:

```bash
$ sudo apt-get install mysql-server libmysqlclient-dev python-mysqldb
$ su - cowrie
$ source cowrie/cowrie-env/bin/activate
$ pip install mysqlclient
```

Previously MySQL-python was used. Only if you run into issues with mysqlclient, try this instead:

```bash
$ pip install MySQL-python
```

21.3 MySQL Configuration

First create an empty database named ‘cowrie’:

```bash
$ mysql -u root -p
CREATE DATABASE cowrie;
```

Create a cowrie user account for the database and grant all access privileges:
GRANT ALL ON cowrie.* TO 'cowrie'@'localhost' IDENTIFIED BY 'PASSWORD HERE';

**Restricted Privileges:**

Alternatively you can grant the cowrie account with less privileges. The following command grants the account with the bare minimum required for the output logging to function:

```
GRANT INSERT, SELECT, UPDATE ON cowrie.* TO 'cowrie'@'localhost' IDENTIFIED BY 'PASSWORD HERE';
```

Apply the privilege settings and exit mysql:

```
FLUSH PRIVILEGES;
exit
```

Next, log into the MySQL database using the cowrie account to verify proper access privileges and load the database schema provided in the docs/sql/ directory:

```
$ cd ~/cowrie/docs/sql/
$ mysql -u cowrie -p
USE cowrie;
source mysql.sql;
exit
```

## 21.4 Cowrie Configuration

Uncomment and update the following entries to ~/cowrie/cowrie.cfg under the Output Plugins section:

```
[output_mysql]
host = localhost
database = cowrie
username = cowrie
password = PASSWORD HERE
port = 3306
debug = false
enabled = true
```

Restart Cowrie:

```
$ cd ~/cowrie/bin/
$ ./cowrie restart
```

Verify That the MySQL Output Engine Has Been Loaded

Check the end of the ~/cowrie/log/cowrie.log to make sure that the MySQL output engine has loaded successfully:

```
$ cd ~/cowrie/log/
$ tail cowrie.log
```

**Example expected output:**

```
...
```
## Confirm That Events are Logged to the MySQL Database

Wait for a new login attempt to occur. Use `tail` like before to quickly check if any activity has been recorded in the `cowrie.log` file.

Once a login event has occurred, log back into the MySQL database and verify that the event was recorded:

```
$ mysql -u cowrie -p
USE cowrie;
SELECT * FROM auth;
```

Example output:

```
+----+--------------+---------+----------+-------------+---------------------+
| id | session      | success | username | password    | timestamp           |
+----+--------------+---------+----------+-------------+---------------------+
| 1  | a551c0a74e06 | 0       | root     | 12345       | 2017-11-27 23:15:56 |
| 2  | a551c0a74e06 | 0       | root     | seiko2005   | 2017-11-27 23:15:58 |
| 3  | a551c0a74e06 | 0       | root     | anko        | 2017-11-27 23:15:59 |
| 4  | a551c0a74e06 | 0       | root     | 123456      | 2017-11-27 23:16:00 |
| 5  | a551c0a74e06 | 0       | root     | dreambox    | 2017-11-27 23:16:01 |
+----+--------------+---------+----------+-------------+---------------------+
```
CHAPTER 22

Using TCP tunneling with Squid

22.1 Prerequisites

- Working Cowrie installation
- Working Squid installation with CONNECT allowed
- (optional) Rate limit and black/white lists in Squid

22.2 Installation

```
$ sudo apt-get install squid
```

22.3 Squid Configuration

See `squid.conf` for an example configuration.

22.4 Cowrie Configuration

Uncomment and update the following entries to `~/cowrie/cowrie.cfg` under the SSH section:

```
forward_tunnel = true
forward_tunnel_80 = 127.0.0.1:3128 forward_tunnel_443 = 127.0.0.1:3128
```

## Restart Cowrie

```
$ cd ~/cowrie/bin/ $ ./cowrie restart
```
Automatically starting Cowrie with supervisord

• Copy the file `cowrie.conf` to `/etc/supervisor/conf/`
NOTE: untested

- Copy the file docs/systemd/system/cowrie.socket to /etc/systemd/system
- Copy the file docs/systemd/system/cowrie.service to /etc/systemd/system
- Examine /etc/systemd/system/cowrie.service and ensure the paths are correct for your installation if you use non-standard file system locations.
- Add entries to etc/cowrie.cfg to listen on ports via systemd. These must match your cowrie.socket configuration:
  
  ```
  [ssh] listen_endpoints = systemd:domain=INET6:index=0
  [telnet] listen_endpoints = systemd:domain=INET6:index=1
  ```

- **Run:** sudo systemctl daemon-reload sudo systemctl start cowrie.service sudo systemctl enable cowrie.service
CHAPTER 25

Indices and tables

- genindex
- modindex
- search