nfsinkhole Documentation

Release 0.1.0

secynic

September 22, 2016

Project Info

1 nfsinkhole **3**
- 1.1 Features . . . 3
- 1.2 Planned Improvements . . . 4
- 1.3 Links . . . 4
- 1.4 Dependencies . . . 5
- 1.5 Installing . . . 5
- 1.6 Service . . . 6
- 1.7 API . . . 7
- 1.8 Contributing . . . 8
- 1.9 Special Thanks . . . 8

2 Contributing **9**
- 2.1 Issue submission . . . 9
- 2.2 Pull Requests . . . 10

3 License **11**

4 Changelog **13**
- 4.1 0.1.0 (2016-08-29) . . . 13

5 apparmor **15**

6 iptables **17**

7 rsyslog **19**

8 selinux **21**

9 service **23**

10 syslog-ng **25**

11 tcpdump **27**

12 utils **29**

13 Library Structure **31**

Python Module Index **37**

nfsinkhole is a Python package for setting up a Unix server as a sinkhole (all protocols/ports to a secondary interface).

CHAPTER 1

nfsinkhole

Warning: This version is considered experimental. Do not attempt to use this library in production until tests via travis and docker are setup, stable, and sufficiently covered.

Attention: You are responsible for rotating log files (/var/log/nfsinkhole*), and syslog forwarding must be configured manually (automation pending).

nfsinkhole is a Python library and scripts for setting up a Unix server as a sinkhole (monitor, log/capture, and drop all traffic to a secondary interface).

The default setup arguments monitor/capture all traffic. Setup arguments are provided to configure protocols, ports, rate limiting, logging, source IP/CIDR exclusions from logging, and optional packet capture.

All sinkhole events are written to /var/log/nfsinkhole-events.log. Optionally, you can enable tcpdump to output packet capture text to /var/log/nfsinkhole-pcap.log if your version of tcpdump supports packet printing; otherwise reverts to /var/log/nfsinkhole.pcap.

1.1 Features

- Simple install script
- Installs as a init.d/systemctl service
- Service modifies iptables on start/stop, no need to persist iptables
- rsyslog and syslog-ng (pending) supported
- RedHat/CentOS 6/7 tested
- Python 2.6+ and 3.0+ supported
- Built-in support for dealing with SELinux/AppArmor
- Packet capture of sinkhole traffic (printed output to log for tcpdump v4.5+)
- Useful set of utilities
- Detailed logging to /var/log/nfsinkhole-*
- Syslog forwarding configuration (pending)
- BSD license

1.2 Planned Improvements

- API/class documentation
- syslog-ng support (currently partially built; unused)
- Tests via travis-ci/docker
- Coverage via coverage.io
- Exception handling overhaul
- Set logging level (currently debug)
- BIND/Microsoft/etc DNS server configuration documentation/examples
- Monitoring use case examples
- Automatic configuration for syslog forwarding
- SIEM parsers/apps/plugins
- Official support/testing for more OS environments
- Support handling exceptions for HIPS and other endpoint security products
- Intelligent handling/handshakes (inspired by iptrap - https://github.com/jedisct1/iptrap)

1.3 Links

1.3.1 Documentation

Release v0.1.0

https://nfsinkhole.readthedocs.io/en/v0.1.0

GitHub master

https://nfsinkhole.readthedocs.io/en/latest

GitHub dev

https://nfsinkhole.readthedocs.io/en/dev

1.3.2 Examples

Pending

1.3.3 Github

https://github.com/secynic/nfsinkhole

1.3.4 Pypi

https://pypi.python.org/pypi/nfsinkhole

1.3.5 Changes

https://nfsinkhole.readthedocs.io/en/latest/CHANGES.html

1.4 Dependencies

OS:

```
iptables (likely already included in base OS)
tcpdump (optional - likely already included in base OS)
```

Python 2.6:

```
argparse
```

Python 2.7, 3.0+:

```
None!
```

1.5 Installing

Attention: The nfsinkhole service, iptables rules, and tcpdump must run as root. You can still use user/virtualenv Python environments, for the library, but ultimately, the core sinkhole will be run as root.

Note: Replace any below occurence of <INTERFACE> with the name of your sinkhole network interface name.

1.5.1 Base OS (pip) – RECOMMENDED

If pip is not installed, you will first need to add the EPEL repo and install:

```
sudo yum install epel-release
sudo yum install python-pip
```

RHEL/CentOS 6/7

Basic:

```
pip install --user --upgrade nfsinkhole
python ~/.local/bin/nfsinkhole-setup.py --interface <INTERFACE> --install --pcap
```

virtualenv:

```
pip install virtualenv
virtualenv nfsinkhole
source nfsinkhole/bin/activate
nfsinkhole/bin/pip install nfsinkhole
nfsinkhole/bin/python nfsinkhole/bin/nfsinkhole-setup.py --interface <INTERFACE> --install --pcap
```

1.5.2 Base OS (no pip)

RHEL/CentOS 6

GitHub - Stable:

```
wget -O argparse.tar.gz https://github.com/ThomasWaldmann/argparse/tarball/master
tar -C argparse -zxvf argparse.tar.gz
cd argparse
python setup.py install --user prefix=
cd ..
rm -Rf argparse
wget -O nfsinkhole.tar.gz https://github.com/secynic/nfsinkhole/tarball/master
tar -C nfsinkhole -zxvf nfsinkhole.tar.gz
cd nfsinkhole
python setup.py install --user prefix=
cd ..
rm -Rf nfsinkhole
python ~/.local/bin/nfsinkhole-setup.py --interface <INTERFACE> --install --pcap
```

RHEL/CentOS 7

GitHub - Stable:

```
wget -O nfsinkhole.tar.gz https://github.com/secynic/nfsinkhole/tarball/master
tar -C nfsinkhole -zxvf nfsinkhole.tar.gz
cd nfsinkhole
python setup.py install --user prefix=
cd ..
rm -Rf nfsinkhole
python ~/.local/bin/nfsinkhole-setup.py --interface <INTERFACE> --install --pcap
```

1.6 Service

Once installed you need to start the nfsinkhole service.

1.6.1 RHEL/CentOS 6

```
sudo service nfsinkhole start
```

1.6.2 RHEL/CentOS 7

```
sudo systemctl start nfsinkhole.service
```

1.7 API

1.7.1 AppArmor

AppArmor documentation:

https://nfsinkhole.readthedocs.io/en/latest/apparmor.html

1.7.2 iptables

iptables documentation:

https://nfsinkhole.readthedocs.io/en/latest/iptables.html

1.7.3 rsyslog

rsyslog documentation:

https://nfsinkhole.readthedocs.io/en/latest/rsyslog.html

1.7.4 SELinux

SELinux documentation:

https://nfsinkhole.readthedocs.io/en/latest/selinux.html

1.7.5 Service

Service (systemd/init.d) documentation:

https://nfsinkhole.readthedocs.io/en/latest/service.html

1.7.6 syslog-ng

syslog-ng documentation:

https://nfsinkhole.readthedocs.io/en/latest/syslog_ng.html

1.7.7 tcpdump

tcpdump documentation:

https://nfsinkhole.readthedocs.io/en/latest/tcpdump.html

1.7.8 Utilities

Utilities documentation:

https://nfsinkhole.readthedocs.io/en/latest/utils.html

1.8 Contributing

https://nfsinkhole.readthedocs.io/en/latest/CONTRIBUTING.html

1.9 Special Thanks

Thank you JetBrains for the PyCharm open source support!

CHAPTER 2

Contributing

2.1 Issue submission

Issues are tracked on GitHub:
https://github.com/secynic/nfsinkhole/issues

Follow the guidelines detailed in the appropriate section below. As a general rule of thumb, provide as much information as possible when submitting issues.

2.1.1 Bug reports

- Title should be a short, descriptive summary of the bug
- Include the OS, Python, and nfsinkhole versions affected
- Provide a context (with code example) in the description of your issue. What are you attempting to do?
- Include the full obfuscated output. Make sure to set DEBUG logging:

```
import logging
LOG_FORMAT = ('[%(asctime)s] [%(levelname)s] [%(filename)s:%(lineno)s] '
    '[%(funcName)s()] %(message)s')
logging.basicConfig(level=logging.DEBUG, format=LOG_FORMAT)
```

- Include sources of information with links or screenshots
- Do you have a suggestion on how to fix the bug?

2.1.2 Feature Requests

- Title should be a short, descriptive summary of the feature requested
- Provide use case examples
- Include sources of information with links or screenshots
- Do you have a suggestion on how to implement the feature?

2.1.3 Testing

Testing code and infrastructure is in progress.

2.1.4 Questions

I am happy to answer any questions and provide assistance where possible. Please be clear and concise. Provide examples when possible. Check the nfsinkhole documentation and the issue tracker before asking a question.

2.2 Pull Requests

2.2.1 What to include

Aside from the core code changes, it is helpful to provide the following (where applicable):

- Unit tests
- Examples
- Sphinx configuration changes in /docs
- Requirements (python2.6.txt, etc)

2.2.2 Guidelines

- Title should be a short, descriptive summary of the changes
- Follow PEP 8 where possible.
- Follow the Google docstring style guide for comments
- Must be compatible with Python 2.6, 2.7, and 3+
- Must not break OS compatibility for RHEL 6/7, CentOS 6/7
- Break out reusable code to functions
- Make your code easy to read and comment where necessary
- Reference the GitHub issue number in the description (e.g., Issue #01)
- When running nosetests, make sure to follow *Testing*

CHAPTER 3

License

Copyright (c) 2016 Philip Hane All rights reserved.

Redistribution and use in source and binary forms, with or without modification, are permitted provided that the following conditions are met:

1. Redistributions of source code must retain the above copyright notice, this list of conditions and the following disclaimer.
2. 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.

THIS SOFTWARE IS PROVIDED BY THE COPYRIGHT HOLDERS AND CONTRIBUTORS "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 COPYRIGHT OWNER OR CONTRIBUTORS 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.

CHAPTER 4

Changelog

4.1 0.1.0 (2016-08-29)

- Initial release

CHAPTER 5

apparmor

TODO

CHAPTER 6

iptables

TODO

CHAPTER 7

rsyslog

TODO

CHAPTER 8

selinux

TODO

CHAPTER 9

service

TODO

CHAPTER 10

syslog-ng

TODO

CHAPTER 11

tcpdump

TODO

CHAPTER 12

utils

TODO

CHAPTER 13

Library Structure

class `nfsinkhole.apparmor.`**`AppArmor`**

The class for managing apparmor policy enforcement, if it is installed.

`disable_enforcement`(*module='usr.sbin.tcpdump'*)

The function for disabling AppArmor enforcement for a module.

Parameters **`module`** – Module in /etc/apparmor.d to disable.

Returns True if disabling enforcement was successful, or False.

Return type Boolean

`enable_enforcement`(*module='usr.sbin.tcpdump'*)

The function for enabling AppArmor enforcement for a module.

Parameters **`module`** – Module in /etc/apparmor.d to enable.

Returns True if enabling enforcement was successful, or False.

Return type Boolean

exception `nfsinkhole.exceptions.`**`BinaryNotFound`**

An Exception for when a binary is not detected.

exception `nfsinkhole.exceptions.`**`IPTablesError`**

An Exception for when a iptables process generates stderr output.

exception `nfsinkhole.exceptions.`**`IPTablesExists`**

An Exception for when iptables rules, related to nfsinkhole, exist.

exception `nfsinkhole.exceptions.`**`IPTablesNotExists`**

An Exception for when iptables rules, related to nfsinkhole, don't exist.

exception `nfsinkhole.exceptions.`**`SubprocessError`**

An Exception for when a generic subprocess generates stderr output.

class `nfsinkhole.iptables.`**`IPTablesSinkhole`**(*interface=None, interface_addr=None, log_prefix="'[nfsinkhole] "', protocol='all', dport='0:65535', hashlimit='1/h', hashlimitmode='srcip, dstip, dstport', hashlimitburst='1', hashlimitexpire='3600000', srcexclude='127.0.0.1'*)

The class for managing sinkhole configuration within iptables.

Parameters

- **interface** – The secondary network interface dedicated to sinkhole traffic. Warning: Do not accidentally set this to your primary interface. It will drop all traffic, and kill your remote access.
- **interface_addr** – The IP address assigned to interface.
- **log_prefix** – Prefix for syslog messages.
- **protocol** – The protocol(s) to log (all traffic will still be dropped). Accepts a comma separated string of protocols (tcp,udp,udplite,icmp,esp,ah,sctp) or all.
- **dport** – The destination port(s) to log (for applicable protocols). Range should be in the format startport:endport or 0,1,2,3,n..
- **hashlimit** – Set the hashlimit rate. Hashlimit is used to tune the amount of events logged. See the iptables-extensions docs: http://ipset.netfilter.org/iptables-extensions.man.html
- **hashlimitmode** – Set the hashlimit mode, a comma separated string of options (srcip,srcport,dstip,dstport). More options here results in more logs generated.
- **hashlimitburst** – Maximum initial number of packets to match.
- **hashlimitexpire** – Number of milliseconds to keep entries in the hash table.
- **srcexclude** – Exclude a comma separated string of source IPs/CIDRs from logging.

create_drop_rule()
The function for writing the iptables DROP rule for the interface.

create_rules()
The function for writing iptables rules related to nfsinkhole.

delete_drop_rule()
The function for deleting the iptables DROP rule for the interface.

delete_rules()
The function for deleting iptables rules related to nfsinkhole.

list_existing_rules(*filter_io_drop=False*)
The function for retrieving current iptables rules related to nfsinkhole.

Parameters **filter_io_drop** – Boolean for only showing the DROP rules for INPUT and OUTPUT. These are not shown by default. This exists to avoid allowing packets on the interface if the service is down. If installed, the interface always drops all traffic regardless of the service state.

Returns Matching sinkhole lines returned by iptables -S.

Return type List

Raises `IPTablesError` – A Unix process had an error (stderr).

class `nfsinkhole.rsyslog.`**RSyslog**(*is_systemd=False*)
The class for managing rsyslog checks and configuration.

Parameters **is_systemd** – True if systemd is in use, False if not (use init.d).

create_config(*prefix='[nfsinkhole] '*)
The function for creating the rsyslog config.

Parameters **prefix** – The log prefix set in iptables.

delete_config()
The function for deleting the rsyslog config.

get_version()
The function for checking the rsyslog version.

Returns rsyslog version string if found, or None.

Return type String

restart()
The function for restarting the rsyslog service.

selinux_associate()
The function for associating the rsyslog config with selinux.

class nfsinkhole.selinux.**SELinux**
The class for managing selinux.

associate(*path*)
The function for associating a file path with selinux

class nfsinkhole.service.**SystemService**(*interface=None, interface_addr=None, log_prefix=”’[nfsinkhole] “’, protocol=’all’, dport=‘0:65535’, hashlimit=‘1/h’, hashlimitmode=’srcip, dstip, dstport’, hashlimitburst=‘1’, hashlimitexpire=‘3600000’, srcexclude=‘127.0.0.1’, pcap=True*)
The class for managing the nfsinkhole init.d/systemd service.

Parameters

- **interface** – The secondary network interface dedicated to sinkhole traffic. Warning: Do not accidentally set this to your primary interface. It will drop all traffic, and kill your remote access.
- **interface_addr** – The IP address assigned to interface.
- **log_prefix** – Prefix for syslog messages.
- **protocol** – The protocol(s) to log (all traffic will still be dropped). Accepts a comma separated string of protocols (tcp,udp,udplite,icmp,esp,ah,sctp) or all.
- **dport** – The destination port(s) to log (for applicable protocols). Range should be in the format startport:endport or 0,1,2,3,n..
- **hashlimit** – Set the hashlimit rate. Hashlimit is used to tune the amount of events logged. See the iptables-extensions docs: http://ipset.netfilter.org/iptables-extensions.man.html
- **hashlimitmode** – Set the hashlimit mode, a comma separated string of options (srcip,srcport,dstip,dstport). More options here results in more logs generated.
- **hashlimitburst** – Maximum initial number of packets to match.
- **hashlimitexpire** – Number of milliseconds to keep entries in the hash table.
- **srcexclude** – Exclude a comma separated string of source IPs/CIDRs from logging.
- **pcap** – Enable packet capture text or raw depending on tcpdump version.’

check_systemd()
The function for checking if systemd is implemented.

Returns A tuple: is_systemd, svc_path.

Return type Tuple (Boolean, String)

create_service()
The function for creating the init.d/systemd service.

delete_service()
The function for deleting the init.d/systemd service.

class nfsinkhole.syslog_ng.**SyslogNG**(*is_systemd=False*)
The class for managing syslog-ng checks and configuration.

Parameters **is_systemd** – True if systemd is in use, False if not (init.d).

create_config(*prefix='[nfsinkhole] '*)
The function for creating the syslog-ng config. (incomplete/unused)

Parameters **prefix** – The log prefix set in iptables.

delete_config()
The function for deleting the syslog-ng config.

get_version()
The function for checking the syslog-ng version.

Returns syslog-ng version string if found, or None.

Return type String

restart()
The function for restarting the syslog-ng service.

selinux_associate()
The function for associating the syslog-ng config with selinux.

class nfsinkhole.tcpdump.**TCPDump**(*sbin='/usr/sbin/tcpdump'*)
The class for managing tcpdump checks.

Parameters **sbin** – Path to tcpdump binary

check_packet_print()
The function for checking if tcpdump/nflog support packet printing.

Returns True if packet printing is supported, or False.

Return type Boolean

get_version()
The function for checking the tcpdump version.

Returns tcpdump version string if found, or None.

Return type String

nfsinkhole.utils.**get_default_interface**()
The function for getting the default Unix network interface address.

Returns The network interface name, or None.

Return type String

nfsinkhole.utils.**get_interface_addr**(*interface=None*)
The function for automatically determining a Unix network interface address.

Parameters **interface** – The network interface name.

Returns The IP address for the interface, or None.

Return type String

nfsinkhole.utils.**popen_wrapper**(*cmd_arr=None*, *raise_err=False*, *log_stdout_line=True*)
The function for subprocess with custom logging output.

Parameters

- **cmd_arr** – Array of command strings to pass to subprocess.Popen().
- **raise_err** – If stderr is encountered, raise SubprocessError.
- **log_stdout_line** – If True, logs each stdout line as a separate log entry. If False, logs all of stdout in a single log entry.

Returns stdout, stderr of the completed subprocess.

Return type Tuple

Raises

- `ValueError` – cmd_arr argument is not provided or is None.
- `TypeError` – cmd_arr argument is not a list.
- `SubprocessError` – The subprocess encountered an error (stderr). raise_err must be True for this.

`nfsinkhole.utils.`**`set_system_timezone`**(*timezone='UTC'*)

The function for setting the system timezone.

Parameters **timezone** – The timezone to set, see /usr/share/zoneinfo/* for options.

Raises `SubprocessError` – One of the processes associated with manual timezone configuration encountered an error.

Python Module Index

n

nfsinkhole, 31
nfsinkhole.apparmor, 31
nfsinkhole.exceptions, 31
nfsinkhole.iptables, 31
nfsinkhole.rsyslog, 32
nfsinkhole.selinux, 33
nfsinkhole.service, 33
nfsinkhole.syslog_ng, 34
nfsinkhole.tcpdump, 34
nfsinkhole.utils, 34

A

AppArmor (class in nfsinkhole.apparmor), 31
associate() (nfsinkhole.selinux.SELinux method), 33

B

BinaryNotFound, 31

C

check_packet_print() (nfsinkhole.tcpdump.TCPDump method), 34
check_systemd() (nfsinkhole.service.SystemService method), 33
create_config() (nfsinkhole.rsyslog.RSyslog method), 32
create_config() (nfsinkhole.syslog_ng.SyslogNG method), 34
create_drop_rule() (nfsinkhole.iptables.IPTablesSinkhole method), 32
create_rules() (nfsinkhole.iptables.IPTablesSinkhole method), 32
create_service() (nfsinkhole.service.SystemService method), 33

D

delete_config() (nfsinkhole.rsyslog.RSyslog method), 32
delete_config() (nfsinkhole.syslog_ng.SyslogNG method), 34
delete_drop_rule() (nfsinkhole.iptables.IPTablesSinkhole method), 32
delete_rules() (nfsinkhole.iptables.IPTablesSinkhole method), 32
delete_service() (nfsinkhole.service.SystemService method), 34
disable_enforcement() (nfsinkhole.apparmor.AppArmor method), 31

E

enable_enforcement() (nfsinkhole.apparmor.AppArmor method), 31

G

get_default_interface() (in module nfsinkhole.utils), 34
get_interface_addr() (in module nfsinkhole.utils), 34
get_version() (nfsinkhole.rsyslog.RSyslog method), 32
get_version() (nfsinkhole.syslog_ng.SyslogNG method), 34
get_version() (nfsinkhole.tcpdump.TCPDump method), 34

I

IPTablesError, 31
IPTablesExists, 31
IPTablesNotExists, 31
IPTablesSinkhole (class in nfsinkhole.iptables), 31

L

list_existing_rules() (nfsinkhole.iptables.IPTablesSinkhole method), 32

N

nfsinkhole (module), 31
nfsinkhole.apparmor (module), 31
nfsinkhole.exceptions (module), 31
nfsinkhole.iptables (module), 31
nfsinkhole.rsyslog (module), 32
nfsinkhole.selinux (module), 33
nfsinkhole.service (module), 33
nfsinkhole.syslog_ng (module), 34
nfsinkhole.tcpdump (module), 34
nfsinkhole.utils (module), 34

P

popen_wrapper() (in module nfsinkhole.utils), 34

R

restart() (nfsinkhole.rsyslog.RSyslog method), 33
restart() (nfsinkhole.syslog_ng.SyslogNG method), 34
RSyslog (class in nfsinkhole.rsyslog), 32

S

SELinux (class in nfsinkhole.selinux), 33

selinux_associate() (nfsinkhole.rsyslog.RSyslog method), 33
selinux_associate() (nfsinkhole.syslog_ng.SyslogNG method), 34
set_system_timezone() (in module nfsinkhole.utils), 35
SubprocessError, 31
SyslogNG (class in nfsinkhole.syslog_ng), 34
SystemService (class in nfsinkhole.service), 33

T

TCPDump (class in nfsinkhole.tcpdump), 34