SHIVA

(Spam Honeypot with Intelligent Virtual Analyzer)

The Honeynet Project

Document version 0.3

1. Introduction

1.1 About

Spam **H**oneypot with **I**ntelligent **V**irtual **A**nalyzer, (i.e. a spam honeypot) written in Python2.7, built on top of Lamson framework. SHIVA is an open-source, high interaction honeypot, and is released under GNU General Public Licence version 3.

SHIVA provides capabilities of collecting and analyzing all spam thrown at it. Analysis of data captured can be used to get information of phishing attacks, scamming campaigns, malware campaigns, spam botnets, etc. SHIVA uses MySQL for storing information.

Below are some of the standout features of SHIVA.

1.1.1 Features

- **Controlled Relay:** SHIVA provides the ability to control the relay part completely. User can enable/disable and set the number of spam to be relayed, in the configuration file. Here is the required section.
- **Open Source:** SHIVA is open source, therefore it is very easy to extend the capabilities. For example, one could easily write a simple module to send the attachments to VirusTotal for further analysis.
- **Identifying Unique Spam:** SHIVA uses fuzzy hashing technique to distinguish between new and already seen spam. This makes it possible to analyze millions of spam and still keeping the database size in check. Python implementation of ssdeep is used.
- **Extracting Information from Spam:** Every spam received passes through the mail parser. SHIVA's mail parser is written to extract all the information that is important. Mail parser extracts source IP, spam's various parts like - to, from, header, subject, body, URLs, attachments, etc. This information is then saved in database, if user has opted for setting up database storage.
- **Supports Authentication:** SHIVA provides more control over the SMTP receiver by adding SMTP authentication feature to it. This way, a user can restrict the access to his SMTP server by setting up credentials.
- **Hpfeeds sharing:** SHIVA also makes sharing the analyzed data easy by adding the Hpfeeds/Hpfriends support. Hpfriends is the social data-sharing platform by The Honeynet Project. Again, Hpfeeds/ Hpfriends can be configured by setting up related options in configuration file.

1.2 General Architecture

SHIVA is divided into two parts - Receiver and Analyzer. In short, the receiver part acts as an open relay SMTP server, collects all the spam thrown at it and dumps them into a local directory. The analyzer, then, picks up the spam and proceeds with analyzing and extracting the information.

1.2.1 Receiver

Receiver part is responsible for collecting spam. Receiver is a lamson project that starts a SMTP listener on specified host address and port. Receiver is configured to dump the incoming spam into a local directory. This local directory is specified in configuration file, under "global" section as "queuepath" variable.

By default, Python's smtpd.py doesn't provide support for authentication. Therefore, some new code was written and much was borrowed from secure_smtpd.py by Benjamin E. Coe. This code added support for AUTH command in smtpd.py. Enabling/ disabling and credentials for SMTP authentication can be managed from configuration file. Options for authentication can be found in "smtpauth" section in configuration file. Figure 1 provides the general code flow of receiver.

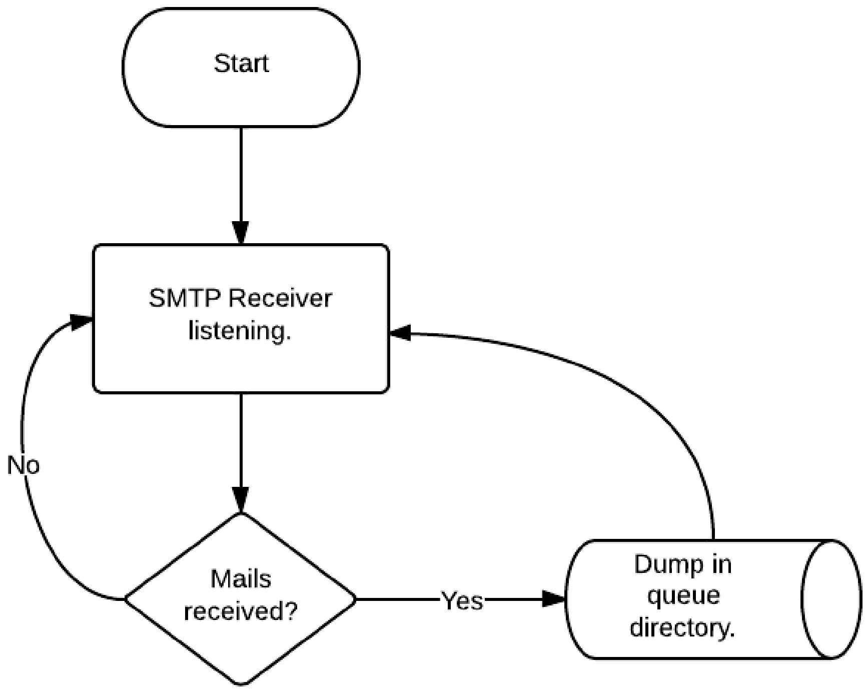

Figure 1. Code flow of receiver.

1.2.1 Analyzer

Analyzer is where things start to get interesting. Analyzer, again is a lamson project. When analyzer is started, it starts waiting for mails in the queue directory. This starts picking up the spam (that were dumped by receiver) and will pass it to SHIVA's custom modules. Normal code flow of Lamson is shown on Figure 2.

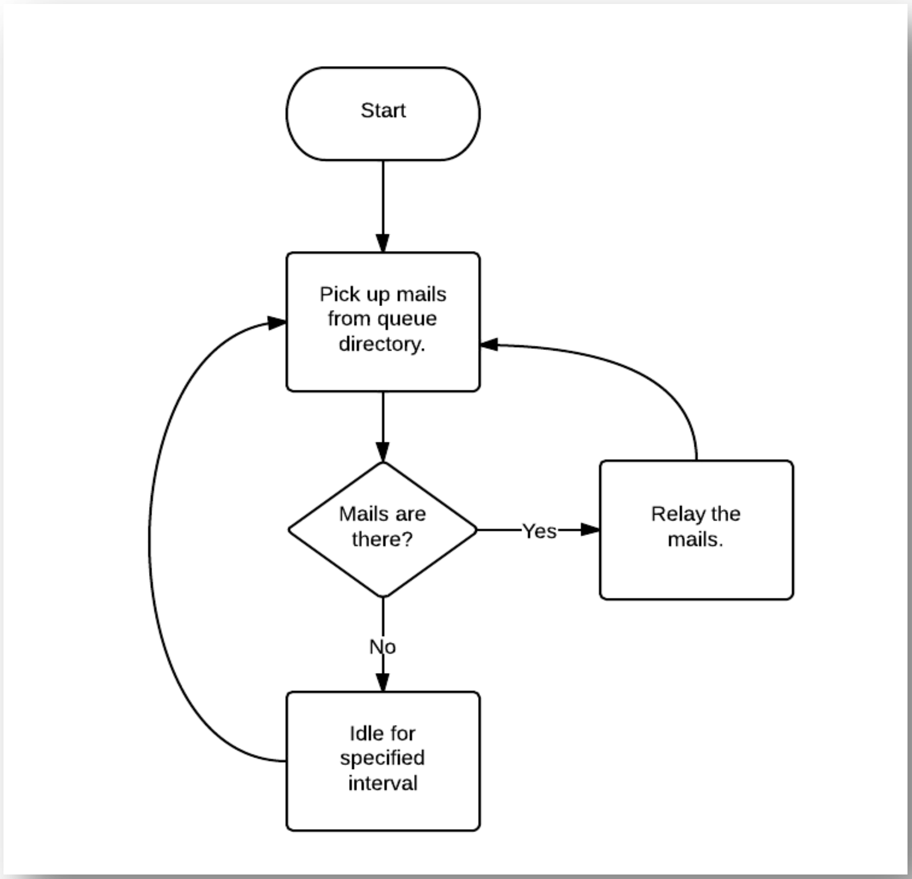
Start
Pick up mails from queue directory.
Mails are there?
Yes
Relay the mails.
No
Idle for specified interval

Figure 2. Normal flow of lamson.

Unlike the above normal flow, after a spam is picked up from queue, we hijack the normal flow of Lamson and, the spam passes through SHIVA's custom modules, instead of being relayed. After a spam is picked up from the directory, it is sent to the Mail Parser module. This module is where all the information is extracted/parsed out of the spam. This extracted information includes, source IP address, spam header, "to" and "from" field, subject of the spam, content or body of the spam, attachments received, URLs, etc. All the parsed data is stored in a dictionary.

After spam is parsed, it is sent to the Shiva Conclude module. This module calculates the SSDEEP i.e. fuzzy hash of the spam and matches it against the pre-stored hashes, stored in list of spam records. If the similarity ratio is more than 85%, spam is considered to be seen before and sent to Process Old module. Else it is sent to Add New Record module.

If the spam is sent to Add New Record module, here a dictionary is made from the parsed data and that dictionary is appended into the list of spam records. After the specified amount of time has passed, these records are saved into the temporary database. The time is specified in the configuration file.

If the spam is sent to Process Old module, then the record is compared against the matched record for any new data in spam. If any new data is found, the corresponding record is updated.

After the time specified for scheduler is passed, the list have spam record is emptied into the temporary database. After this, Main DB module is called, which co-relates the data between temporary database and main database. By co-relation here, we mean matching if the data that is in temporary database, exist in our main database or not. If it exist, we check if any update is required. If it doesn't exist, we save that record in our main database.

The figure for this code flow can be found in Github repository under the "*/docs*" directory.

1.3 Use Cases

SHIVA , other than being used as traditional open but controlled relay honeypot, can be used for other purposes too. Listed below are some of the possible usage scenarios:

- **Collecting Spam**: SHIVA can be used as an SMTP server that'll dump all the spam that it receives, into a local directory. If we start only the receiver part, then SHIVA will act as an SMTP server, and all mails will be dumped in queue directory.
- **Analyzing Stored Spam**: SHIVA can also be used to analyze the spam that might've been collected from various sources. All we need to do is to transfer all the spam into the queue folder and start the analyzer part.
- **Hpfeeds Sharing**: SHIVA can be easily configured for sharing data to hpfeeds. There's a dedicated section in configuration file that deals with all the things needed to set up hpfeeds sharing. It can be primarily used for hpfeeds sharing only, by disabling local database storage.
- **Client-Server Setup -** With a little trouble, SHIVA can be used to setup up a client-server infrastructure. In this setup, there'll be multiple sensors collecting the unique spam and sending it to the master node. This master node will further analyze those spam and will save the data. Client nodes can share the spam via Hpfeeds, in form of raw spam files.

1.4 Other Spam Honeypots

Open relay honeypots include Jackpot, smtpot.py and spamhole and The Bubblegum Proxypot. (Source Wikipedia).

1.5 Authors

- **Sumit Sharma** <sumit.iips@gmail.com>
- **Rahul Binjve** <rahulbinjve@gmail.com> @RahulBinjve

2. Setting up SHIVA

Installing SHIVA has been made pretty straightforward and simple. It is as easy as running a single bash script, given user has required permissions and prerequisites installed.

2.1 Prerequisites

We expect following packages to be installed on system before user starts the installation procedure.

- Debian based OS (Preferably Ubuntu or Mint Linux)
- Python2.7
- exim4-daemon-light
- g++
- python-virtualenv
- python-dev
- libmysqlclient-dev
- mysql-client (optional, only if user wants to save spam data in databases.)
- mysql-server (optional, only if user wants to set up MySQL database in same machine.)
- phpmyadmin (optional, only if user wants to monitor databases, the GUI way.)

2.2 Installation

2.2.1 Preparing System

On a freshly Ubuntu system, user can install all the required packages by executing this command (assuming Python is pre-installed).

```
$ sudo apt-get install python-dev exim4-daemon-light g++ python-virtualenv libmysqlclient-dev
```

2.2.2 Getting SHIVA (the Github way)

Clone the SHIVA's Github repository in the folder where SHIVA is to be set up.

```
$  git clone https://github.com/shiva-spampot/shiva.git shiva-installer
```

User can confirm that repository has been cloned properly by checking the content of *"shiva-installer"* folder.

```
$ cd shiva-installer

$ ls
```

2.2.3 Installing

SHIVA can be easily installed by executing the bash installer script.

- The installer can be run as:

```
$ ./install.sh
```

- After dependencies have been checked, user will be asked to confirm if he wants to setup local databases or not.

```
Do you wish to setup local databases for storing spam?

[Y]es/ [N]o...
```

If the user inputs "Yes", appropriate instructions will be printed on console to set up databases.

Installation will proceed further, then. If everything goes well, everything will be installed in *"shiva"* folder.

2.2.4 Setting up database

If user wants to store spam data, SHIVA needs to setup two databases. This section deals with setting up MySQL databases.

Note: The following steps assume that user is setting up MySQL server on the same machine. However, this practice is discouraged.

Before setting up databases, run the following command to install the dependencies:

```
$ sudo apt-get install mysql-server mysql-client
```

After the above packages have been installed, follow these steps to setup databases:

- Edit the configuration file located at *"shiva/shiva.conf"* and provide the necessary parameters required under the *"database"* section.
- Run the python script that'll creates necessary databases and tables.

```
$ cd shiva

$ python dbcreate.py
```

- If script exits without any error, databases have been created.

This can be confirmed by connecting to mysql server and checking the available databases by using *"show databases;"* command. The output should contain "Shiva" and "Temp".

2.2.5 Setting up exim

For relay, SHIVA requires exim to be installed.

Since, by default exim starts listening on port 25, user needs to configure it to listen on port 2500. Exim can be configured by running the bash script, included in installer package. Follow the steps below:

```
$ cd shiva

$ sudo sh setup_exim.sh
```

This will configure exim and restart it.

Note: User should have "*exim4-daemon-light*" package installed. If user wish to use any other Mail Transfer Agent (MTA) than exim4, feel free to do so, as relaying mail is totally an outsider task. Therefore, it shouldn't cause any problem in SHIVA's normal functioning. If any other MTA is used, write to us explaining the procedure. We'll make sure to add it in documentation.

3. Configuration

Configuring SHIVA is a piece of cake, credits to ConfigParser, a Python library. Everything that needs to be configured, is in "shiva/shiva.conf" file. The file is divided into various sections, namely:

- global
- receiver
- analyzer
- database
- hpfeeds

Below description has been provided about each option.

3.1 Global

Configuration(s) that applies for both, analyzer and receiver.

- *queuepath* - Path where all the spam will be dumped and retrieved for analyzing. This will be updated by installer script itself.

Note: Even though the 'queuepath' can be changed, currently it is recommended to use default settings only.

3.2 Receiver

SHIVA receiver's configuration.

- *listenhost* - The host where to start SMTP receiver on, usually public IP. "localhost", by default.
- *listenport* - The port to listen on for incoming SMTP connections. 25, by default.
- *sensorname* - Name of the sensor. "shiva", by default.

3.2.1 SMTP AUTH

If user wants SMTP AUTH in SHIVA, edit these options.

- *authenabled* - Boolean value to enable/disable SMTP AUTH. "False", by default.
- *smtpuser* - Username for SMTP AUTH.
- *smtppasswd* : Password for SMTP AUTH.

3.3 Analyzer

All the options for SHIVA analyzer.

- *relay* : enable/disable relaying of spam.
- *globalcounter* : Number of total spam to be relayed, in a specific duration.
- *individualcounter* : Number of times an indivisual spam is to be relayed, in a specific duration.
- *relayport* : 2500 The port on which MTA is listening.
- *relayhost* : The host on which MTA is listening.
- *undeliverable_path* : Path where distorted spam will be dumped. Will be updated by installer script.
- *schedulertime* : Duration (in minutes) to be passed to shivascheduler module.
- *rawspampath* : Path where raw spam samples will be dumped. Will be updated by installer script.
- *attachpath* : Path to dump attachments . Will be updated by installer script.
- *inlinepath* : Path to dump inline attachments. Will be updated by installer script.

3.4 Database

Database related configurations.

- *localdb* - Boolean value to enable/disable database storage.
- *host* - MySQL host to connect. "localhost", by default.
- *user* : MySQL username. "root", by default.
- *password* : MySQL password. "password", by default.

3.5 Hpfeeds

Options for enabling Hpfeeds sharing.

- *enabled* - Boolean value to enable/disable hpfeeds.
- *host* - Hpfeeds host.
- *port* - Hpfeeds port.
- *ident* - Ident for hpfeeds.
- *secret* - Secret key for hpfeeds.

4. Running SHIVA

For running SHIVA, open two terminals. One for receiver and another one for analyzer. To start SHIVA, we need to activate the respective virtual environments and starting the lamson instances.

4.1 Shiva Receiver

Shiva Receiver is the first half of SHIVA that starts a SMTP server on host and port that user has specified in configuration file. To start the receiver, follow these steps. (**Note:** Requires root rights.)

```
$ sudo su

# cd shiva/ShivaReceiver/
```

- Activate virtual environment.

```
# source bin/activate
```

- Now, the prompt should be similar to this

```
(ShivaReceiver) #
```

- Now, we'll start lamson that'll start SMTP server.

```
(ShivaReceiver) # cd Receiver/

k(ShivaReceiver) # lamson start
```

To check if lamson started correctly, user can either check the log files or by using *'netstat'* Linux command.

- Checking logs

```
(ShivaReceiver) $ cd logs/

(ShivaReceiver) $ head lamson.log
```

This will output something like "SMTP Receiver successfully started."

Or, the 'netstat' command.

```
(ShivaReceiver) $ sudo netstat -natp | grep 25
```

This command should show *"python"* listening on port 25.

4.2 Shiva Analyzer

Shiva Analyzer is the part of SHIVA where all the analyzing work is done. Analyzer starts lamson and waits for spam to arrive in "queue" folder. Queue's path is defined in "shiva.conf" similar to this

```
[global]

queuepath : /path/to/shiva/queue/
```

- To start the receiver, follow these steps.

```
$ cd shiva/ShivaAnalyzer/
```

- Activate virtual environment.

```
$ source bin/activate
```

- Now, the prompt should be similar to this

```
(ShivaAnalyzer) $
```

- Now start lamson that'll start queue receiver.

```
(ShivaAnalyzer) $ cd Analyzer/

(ShivaAnalyzer) $ lamson start
```

To check if lamson started correctly, user can check the log files.

- Checking logs

```
(ShivaReceiver) $ cd logs/

(ShivaReceiver) $ head lamson.log
```

This will output something like "Queue receiver started on queue dir /path/to/shiva/queue/."

4.3 Checking Logs

Logs for receiver can be found at

```
$ cd /path/to/shiva/ShivaReceiver/Receiver/logs/

$ ls

lamson.err  lamson.log  lamson.out  clearlogs.sh
```

- lamson.err - All the errors and relay logs are stored in this file.
- lamson.log - All the logs generated by SMTP receiver.
- lamson.out - Anything supposed to be printed on console by program is written in this file.
- clearlogs.sh - Bash files that'll delete the log files.

Similarly for analyzer, logs are at

```
$ cd /path/to/shiva/ShivaReceiver/Receiver/logs/

$ ls

lamson.err  lamson.log  lamson.out  clearlogs.sh
```

- lamson.err - All the errors and relay logs are stored in this file.
- lamson.log - All the logs generated while analyzing spam.
- lamson.out - Anything supposed to be printed on console by program is written in this file.
- clearlogs.sh - Bash files that'll delete the log files.

4.4 Stopping SHIVA

Receiver and Analyzer, both can be stopped by stopping their respective lamson instances. Before doing so, user needs to activate their respective instances.

Therefore, to stop Receiver, follow these steps:

```
$ cd shiva/ShivaReceiver

$ source bin/activate

(ShivaReceiver) $ cd Receiver/

(ShivaReceiver) $ lamson stop
```

Similarly, for stopping Analyzer:

```
$ cd shiva/ShivaAnalyzer

$ source bin/activate

(ShivaAnalyzer) $ cd Analyzer/

(ShivaAnalyzer) $ lamson stop
```

5. General Problems and Precautions

This section deals with the general troubleshooting tips and things to keep in mind when setting up a honeypot.

5.1 Known Issues and Troubleshooting

- **Unable to relay spam**

- Possible reasons might be:

  - Residential or Dynamic IP Address - Almost all the major mail providers don't allow mails from dynamic or residential IP addresses. To relay a spam/mail, user will need a static IP address.
  - "Exim" Errors - Since, exim is responsible for relaying of spam, we might want to check the exims logs. Exim logs can be found at "/var/logs/exim4/". Make sure exim is configured as stated here.

- **"MySQL server has gone away"**

- By default, MySQL server keeps connection alive for 8 hrs. If no activity happens during this duration, it will close the connection. This limit can be increased by changing value of "*wait_timeout*" variable and restarting mysqld service.

  - Edit "/etc/mysql/my.cnf"

```
wait_timeout=86400
```

  - Restart mysqld service.

```
$ sudo service mysql restart
```

Note: More information can be found here.

- **Unable to connect to SMTP receiver?**

- It is advisable to restart the SMTP Receiver after 24 hours or so. Instructions for stopping and starting receiver can be found here and here, respectively.

- **Too many connections showing up in *netstat* output?**

  - We've seen cases of *"netstat"* showing too many connections as "ESTABLISHED", even though they're not. We're yet to determine the cause of this problem. The solution is to stop the receiver end, wait for few minutes and start the receiver.

- **Too many spam in queue folder?**

  - Currently, analyzer could analyze spam at a certain speed. If the amount of spam that you're receiving is too high, the size of queue directory will increase and causes adverse effect on analyzer's speed. Therefore, in such scenario, it is advised to stop the receiver for some time, i.e. till all the spam are analyzed.

5.2 General Security Practices

Coming soon.

6. Frequently Asked Questions

- *"SHIVA doesn't work or keeps on breaking while analyzing spam"?*

SHIVA is in infant stage, therefore, we believe that this is a normal scenario. We're constantly working and trying to make it more stable and user-friendly. If you cannot make SHIVA run, please read this documentation carefully. If that doesn't help, refer the log files and google if you see any error.

Even then if you can't make SHIVA work, blame it on the complexity of setting up a honeypot. *wink*

- *Deactivating virtual environment?*

Python's virtual environments can be simply deactivated by

```
$ deactivate
```

- *There's unusual amount of storage used by SHIVA, why?*

SHIAV produces lots of logs. There are receiver logs, analyzing logs, hpfeeds logs, relay logs, and what not. If you see high storage usage, you can go to logs folder and run the '*clearlogs.sh*' script. Log files path information can be found here.

- *How to confirm if both receiver and analyzer are running?*

- Since, receiver and analyzer are two different parts, if everything's working fine, there must be two '*Lamson'* instances running on your system. This can be confirmed by using *'ps',* another great command.

```
$ ps -el | grep -i lamson
```

This command will show the Lamson processes that are running in system. If this shows 2 lamson instances running, you're good to go, most probably.

- *Adding support for Virus Total/ Cuckoo sandbox API?*

Due to the simple design of SHIVA, a small module could be easily written, to send the attachments and URLs received in a spam to Virus Total or Cuckoo sandbox for further analysis.

- *What are the recommended system specification?*

- These are the recommended system specifications:

    - Processor - Intel Dual Core or higher
    - OS - Debian-based. Tested on Ubuntu 12.04/13.04 and Mint Linux 15.
    - RAM - 2 Gb or more
    - Hard disk - 5 Gb or more.
    - Graphic card - Meh, kidding?

- *I found a bug. Now?*

- If you found a bug, report it to us. For contact info, check authors' info. If you've patch for it, that's even better.

- *My IP has been blocked by Google/ Yahoo!, Hotmail, etc. What do I do now?*

- You got your IP blocked? Well, that happens more often, than not. If your IP is blocked, there's nothing that we could do, actually. However, we'll suggest to set relay counters to 0 (i.e. stop relaying) for some days and wait for your IP to get unblocked (if it does). And when you restart relaying, keep the relay counters low.

- How to contribute?

- The project is in infant stage, so we need contribution from community. Actually, loads of contribution. If you've a bug, report it to us, For contact info, check authors' info. If you've a patch or you've added some cool feature, clone our github repository, send us a pull request. We'll be more than happy to merge your patches.

Please note that while submitting patches, make sure patch follows PEP 8 -- Style Guide for Python Code.

7. References

Following documents were referred while preparing this document.

- https://write-the-docs.readthedocs.org/en/2013/writing/oss.html
- http://www.honeynet.org/files/KYT-Glastopf-Final_v1.pdf
- http://docs.cuckoosandbox.org/en/latest/