Proactive Incident Detection

Toolset, Document for students

September 2014
About ENISA

The European Union Agency for Network and Information Security (ENISA) is a centre of network and information security expertise for the EU, its member states, the private sector and Europe’s citizens. ENISA works with these groups to develop advice and recommendations on good practice in information security. It assists EU member states in implementing relevant EU legislation and works to improve the resilience of Europe’s critical information infrastructure and networks. ENISA seeks to enhance existing expertise in EU member states by supporting the development of cross-border communities committed to improving network and information security throughout the EU. More information about ENISA and its work can be found at www.enisa.europa.eu.

Acknowledgements

Contributors to this report

We would like to thank all our ENISA colleagues who contributed with their input to this report and supervised its completion, especially Lauri Palkmets, Cosmin Ciobanu, Andreas Sfakianakis, Romain Bourgue, and Yonas Leguesse. We would also like to thank the team of Don Stikvoort and Michael Potter from S-CURE, The Netherlands, Mirosław Maj and Tomasz Chlebowski from ComCERT, Poland, and Mirko Wollenberg from PRESECURE Consulting, Germany, who produced the second version of this documents as consultants.

Agreements or Acknowledgements

ENISA wants to thank all institutions and persons who contributed to this document. A special ‘Thank You’ goes to the following contributors: Anna Felkner, Tomasz Grudzicki, Przemysław Jaroszewski, Piotr Kijewski, Mirosław Maj, Marcin Mielniczek, Elżbieta Nowicka, Cezary Rzewuski, Krzysztof Silicki, Rafał Tarłowski from NASK/CERT Polska, who produced the first version of this document as consultants and the countless people who reviewed this document.

Contact

For contacting the authors please use CERT-Relations@enisa.europa.eu
For media enquires about this paper, please use press@enisa.europa.eu.
Legal notice

Notice must be taken that this publication represents the views and interpretations of the authors and editors, unless stated otherwise. This publication should not be construed to be a legal action of ENISA or the ENISA bodies unless adopted pursuant to the Regulation (EU) No 526/2013. This publication does not necessarily represent state-of-the-art and ENISA may update it from time to time.

Third-party sources are quoted as appropriate. ENISA is not responsible for the content of the external sources including external websites referenced in this publication.

This publication is intended for information purposes only. It must be accessible free of charge. Neither ENISA nor any person acting on its behalf is responsible for the use that might be made of the information contained in this publication.

Copyright Notice

© European Union Agency for Network and Information Security (ENISA), 2013

Reproduction is authorised provided the source is acknowledged.
# Table of Contents

1 What Will You Learn

2 Exercise Task
   2.1 Task 1 Setting up AbuseHelper
   2.2 Task 2 Working with Abusehelper

3 Conclusion
1 What Will You Learn

In this exercise you will learn how to set up and work with AbuseHelper. AbuseHelper is an Open Source Software designed to help CERT/CSIRT organization with consolidating and using information feeds.

- You will learn how to install and configure all parts of the AbuseHelper toolset
- You will learn how to take the application set into production
- You will learn how to identify useful information and how to handle it

2 Exercise Task

The instructor will give an introduction to the topic of information consolidation and background details to the development and technology used in AbuseHelper. Mandatory for this exercise is a working internet connection as the feeds will be pulled in from online sources. All necessary parts for running the application set are available on the Virtual Image (/usr/share/trainee/14_PID/).

You will find helpful information in the documents in the References folder.

2.1 Task 1 Setting up AbuseHelper

Fill this table:

<table>
<thead>
<tr>
<th>No.</th>
<th>Required</th>
<th>Answer</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>XMPP user names</td>
<td></td>
</tr>
<tr>
<td></td>
<td>XMPP user password</td>
<td></td>
</tr>
<tr>
<td></td>
<td>Name of lobby room</td>
<td></td>
</tr>
<tr>
<td></td>
<td>Output: <code>botnet status /var/lib/ah2/production</code></td>
<td></td>
</tr>
<tr>
<td></td>
<td>Have you received alert mails?</td>
<td></td>
</tr>
</tbody>
</table>

1. Ejabber Daemon

The Ejabber Daemon is the communication channel for AbuseHelper. It is essential for the exchange of information between bots and users. The daemon has been installed from the Ubuntu repository but some basic configuration changes have to be applied.

`sudo /etc/init.d/ejabberd start # Start the Jabber service`

---

1 [Ejabberd Installation and Operation Guide](#)
sudo ejabberdctl register abusehel localhost exercise # register a user for the bots (username host password)

sudo ejabberdctl register trainee localhost exercise # register a user (username host password)

sudo vi /etc/ejabberd/ejabberd.cfg # open the ejabberd configuration file and edit the following lines

max_user_sessions 100 # maximum sessions for a single user

s2s_default_policy deny # deny server to server communication

%{shaper, c2s_shaper}, # search for and comment out the default shaper configuration

Figure 1: ejabberd.cfg shaper configuration

{mod_muc, [
  %{host, "conference.@HOST"},

  {access, muc},

  {access_create, muc},

  {access_persistent, muc},

  {access_admin, muc_admin},

  {max_usersAdminThreshold, 20}, # add this entry

  {max_user_conferences, 1000}, # add this entry

  {max_users, 500}
]}
Figure 2: ejabberd.cfg shaper configuration

```
Figure 2: ejabberd.cfg shaper configuration
```

2. **AbuseHelper**

   The next step will be to install the AbuseHelper applications and create the basic configuration.

   ```
sudo useradd -m abusehel # add a system user for AbuseHelper
sudo mkdir -p /var/lib/ah2 # create the working directory
sudo chown root:abusehel /var/lib/ah2 # ownership of the working directory
sudo chmod 0750 /var/lib/ah2 # directory access rights set to read, write

cd /usr/share/trainee/14_PID/adds/abusehelper/ # change your current directory (trainee for the students)
sudo python setup.py install # run the AbuseHelper setup script

cd /usr/local/lib/python2.7/dist-packages/abusehelper # change directory
sudo python contrib/confgen/confgen.py /var/lib/ah2/production # start the configuration script

   Enter the following information:

   - **XMPP username**: abusehel@localhost # as defined during user registration
   - **XMPP password**: exercise # you will be asked to enter this twice
   - **XMPP lobby channel**: abusehelper # this is the initial channel to connect to when starting the Jabber client

   - **Configure mailer? Yes** # let AbuseHelper send alert mails

   - **SMTP host**: localhost # use the local MTA for delivery
   - **SMTP port**: 25 # use the standard SMTP port
   - **SMTP auth user**: no auth # no authentication necessary
   - **Mail sender**: abusehelper@localhost # mail sender address

   ```
sudo chown -R root:abusehel /var/lib/ah2/production # access rights have to be corrected after the configuration script
sudo chmod 0750 /var/lib/ah2/production # see above
```
sudo chmod g+w /var/lib/ah2/production/archive # see above

sudo chown abusehel /var/lib/ah2/production/log # this directory has been added and must be owned by the abusehel system user for logging

sudo chown abusehel /var/lib/ah2/production/state # see above

sudo vi /var/lib/ah2/production/startup.py # open the startup script and check the entries made by means of the confgen script

Insert this line after `service_room=service_room`, in the `def basic` section:

```python
xmpp_ignore_cert = True, # this deactivates checking ssl certificates
```

Comment out the following line in the `def configs` section:

```python
# yield basic(“roomgraph”)
```

Configure the mail recipient in the runtime.py file:

```bash
sudo vi /var/lib/ah2/production/runtime.py
```

Change the recipient from someone@example.com to trainee@localhost
3. Start AbuseHelper
Now we are ready to start the AbuseHelper application along with the basic bots.

```
sudo su - abusehel -s /bin/bash # change to the abusehel system user
botnet start /var/lib/ah2/production # start the bots defined in the startup.py script
botnet status /var/lib/ah2/production # ask for the status, at least one instance should be running
botnet stop /var/lib/ah2/production # stop the AbuseHelper bots
```

Logs can be found in these directories:
```
/var/lib/ah2/production/log/
/var/log/ ejabberd/
```

To enable logging functionality for every bot (logs can be found from /var/lib/ah2/production/log) uncomment the lines outlined below in the picture.
4. Start Jabber clients

Communication with AbuseHelper and gathering information from the bots is mainly done by means of Jabber clients. There are several Jabber clients installed on the VM, you should at least try the following ones:

- **Psi+**
  Graphical client, you will have to trust the certificate presented by the Jabber service manually.

This screenshot shows the service discovery feature:
Roomreader is a command line client that comes with AbuseHelper. To use it, you can run the following command:

```
roomreader --xmpp-ignore-cert trainee@localhost abusehelper
```

Figure 7: Psi+ service discovery

Figure 8: Roomreader
2.2 Task 2 Working with Abusehelper

1. Making yourself familiar with AbuseHelper
   First step will be to watch the different subrooms and identify the information flow. The table at the end of this section helps to structure and evaluate the learning process.

2. Carry on and include additional data feeds
   In this task the students should identify and describe the data feeds in the /usr/share/lib/python2.7/dist-packages/abusehelper/contrib section and document which to include. Afterwards they should configure the bots in the startup.py and runtime.py files (error messages will be logged to the bot files in the /var/lib/ah2/production/log/ folder).

3. Filter information feeds
   There are different ways to filter the incoming information to be more relevant to your organisations infrastructure.
   Start with dshield and open the runtime.py. You will find an entry regarding the ASN. Change the (Autonomus System Numbers) ASN to your organisation’s network(s). Edit runtime.py (/var/lib/ah2/production/runtime.py) to filter ASN numbers. A list linking ASN to organisations can be found here. The functionality of this filter mechanism is implemented in the dshield bot itself.
   Sanitizers take the raw data provided by the bots, clean it according to the configuration and deliver it into the abusehelper.sources room. Examples for sanitizer scripts are available in /var/lib/ah2/production/custom/. These can be easily adapted for other bots. These fields/tags can be used in rules. Create a sanitizer script for one of the bots from contribution and modify it to add the “ENISA” tag to the output.
   You can write rules to filter output. First tweak the def _mail section in the runtime.py to use abusehelper.sources as data input. Then add a customer definition to send all data tagged with “ENISA” to trainee@localhost

Fill the table with the requested information:

<table>
<thead>
<tr>
<th>No.</th>
<th>Question</th>
<th>Answer</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>Which feeds are standard?</td>
<td></td>
</tr>
<tr>
<td>2</td>
<td>Which information do these deliver?</td>
<td></td>
</tr>
<tr>
<td>3</td>
<td>Where are additional feed bots available?</td>
<td></td>
</tr>
<tr>
<td>4</td>
<td>Integrate the Arbor SSH bot</td>
<td>startup.py: yield basic('arborssh','abusehelper.contrib.arbor.ssh')</td>
</tr>
</tbody>
</table>
<pre><code>                                       | runtime.py: yield source('arborssh')                                   |
</code></pre>
### Conclusion

Finishing this exercise you will have learned the following:

- Installation of Abusehelper
- Base configuration of AbuseHelper and default bots
- Evaluating and integrating bots from the contribution folder
- Filtering of input and output information with rules
- Accessing AbuseHelper information

The provided information should give you a starting point for the evaluation/implementation of AbuseHelper in your organization.