10. Exercise: Automation in Incident Handling

Main Objective	The purpose of this exercise is to develop students' abilities to create custom scripts and filters dealing with large amounts of data like IP addresses. After completing the exercise students should be able to extract useful information from bulk data, even in non-standard formats.	
Targeted Audience	Incident handlers and technical staff. This exercise does not require experience in incident handling. It can be used for experienced as well as future CERT members. Basic knowledge of Linux shell commands, text manipulation tools and/or programming is required.	
Total Duration	1 hour, 45 minutes	
Time Schedule	Introduction to the exercise	15 min.
	Task 1: Locating unique interesting hosts	20 min.
	Task 2: Geolocation	30 min.
	Task 3: Looking further	30 min.
	Summary of the exercise	10 min.
Frequency	Once a year, for new team members or members reassigned to technical tasks.	

10.1 GENERAL DESCRIPTION

Sometimes information about an incident, particularly a wide-spread incident, is received in bulk – containing not just data about your networks but from all networks. This can be the case when a site under a DDoS attack shares its logs without time to sort and separate them for individual ISPs, look for contacts, etc. Having one-to-many distribution channels at hand, such as mailing lists, they can efficiently publish information for everyone to analyse.

On the other hand, sometimes you have plenty of information collected from your own sources that you wish to share with others, distributing it on a need-to-know basis. An example can be logs from IPS systems, early warning systems, etc. While you observe attacks from all around the world, you may have a few interested parties who want to receive and handle reports about their networks. In such cases you need to weed out the information.

10.2 Tools for preparing the exercise

It is recommended that the students use a Linux shell with standard tools such as grep, awk, wc, etc. They may also choose to use Perl or other tools entirely or even an environment with which they feel more comfortable.

The students will find all necessary files for analysis, as well as Linux tools, on the Virtual Image. The logs file lives in `/usr/share/trainer/10_AIH/24022007.txt`. Should they choose to use another environment, they will need to transfer the files using flash drives or other media.

Internet connectivity must be provided in order to conduct the exercise.

10.3 EXERCISE COURSE

The solutions given here are just examples, using tools listed in the students' book and available on the Virtual Image.

10.4 Introduction to the exercise

Introduce the students to the exercise. Ask them about possible situations when automation may be useful or even necessary during incident handling (other than those listed in the introduction).

Possible answers:

- Reports from automated systems such as SpamAssassin or in-house early warning/IDS systems; and
- Spamboxes (header analysis).

10.4.1 Task 1 Locating unique interesting hosts

Assuming that the attack consists only of UDP packets and that all UDP packets were from attacking hosts, there are 142 unique source addresses.

Proposed solution:

```
$ grep UDP 24022007.txt | awk '{print $5}' | awk -F: '{print $1}' | sort -u | wc -l
```

Explanation:

`grep UDP 24022007.txt`	Limit the log to lines regarding UDP packets.
`awk '{print $5}'`	Print the 5th field of the file (src_address:port)
`awk -F: '{print $1}'`	Print the part before colon (first field, ":" separates)
`sort -u`	sort file, output only unique lines
`wc -l`	count lines in the file

10.4.2 Task 2 Geolocation

The IP to AS *whois* provided by Team Cymru supports bulk queries using netcat. In order to use that functionality you need to create a text file with a little extra formatting (note the instructions from http://www.team-cymru.org/Services/ip-to-asn.html#whois).

Proposed solution:

```
$ echo -e "begin\ncountrycode" > 1.tmp
$ grep UDP 24022007.txt | awk '{print $5}' | awk -F: '{print $1}' | sort -u >> 1.tmp
$ echo "end" >> 1.tmp
$ netcat whois.cymru.com 43 < 1.tmp > 2.tmp
$ grep " PL " 2.tmp
$ grep " TR " 2.tmp
```

Explanation:

`echo -e "begin\ncountrycode" > 1.tmp`	create file "1.tmp", write:
	`begin`
	`countrycode`
	to that file
	This is formatting required by the AS-to-IP whois server with bulk queries. The second line needs to be added in order to include the country codes in the output.

`grep …`	generate list of unique sources (see above), append to file
`echo` "end" >> 1.tmp	append the word 'end' to the end of the file – required by the whois server
`netcat whois.cymru.com 43 < 1.tmp > 2.tmp`	send the contents of the file '1.tmp' to whois.cymru.com port 43 tcp, save output into '2.tmp'
`grep " PL " 2.tmp`	print lines containing string ' PL'

Important: Note that sending individual requests with each IP address in the loop, although technically possible, is strongly discouraged by the owners of the server.

Results:

Note that the results may change as IP addresses may occasionally be reassigned. This is the expected output for .PL. The format of the output is as follows:

Column	Description
1	Autonomous System Number
2	IP Address
3	Country Code
4	Autonomous System Description

```
8308    | 193.59.201.24     | PL | NASK-COMMERCIAL NASK
8308    | 193.59.201.28     | PL | NASK-COMMERCIAL NASK
5617    | 194.204.158.242   | PL | TPNET Polish Telecom_s commercial IP network
5617    | 194.204.159.17    | PL | TPNET Polish Telecom_s commercial IP network
5617    | 194.204.159.19    | PL | TPNET Polish Telecom_s commercial IP network
41079   | 195.114.1.252     | PL | SUPERHOST-PL-AS SuperHost.pl s.c.
5617    | 195.116.213.34    | PL | TPNET Polish Telecom_s commercial IP network
6885    | 195.128.113.229   | PL | RSK-ASN RSK.PL Autonomous System
8308    | 195.187.244.4     | PL | NASK-COMMERCIAL NASK
8308    | 195.187.244.8     | PL | NASK-COMMERCIAL NASK
8308    | 195.187.245.44    | PL | NASK-COMMERCIAL NASK
8308    | 195.187.245.51    | PL | NASK-COMMERCIAL NASK
5617    | 195.205.249.130   | PL | TPNET Polish Telecom_s commercial IP network
8364    | 212.126.28.169    | PL | POZMAN-COM
8286    | 212.14.1.62       | PL | ACI-AS ACI Autonomous System
13119   | 212.14.63.195     | PL | ACI-AS ACI Autonomous System
16283   | 212.191.132.126   | PL | LODMAN-AS2 Metropolitan Area Network LODMAN
5617    | 212.244.52.161    | PL | TPNET Polish Telecom_s commercial IP network
20804   | 213.172.174.70    | PL | ASN-TELENERGO EXATEL S.A. Autonomous System
8938    | 213.218.118.26    | PL | ENERGIS-IP Energis Polska IP Network
5617    | 217.98.63.165     | PL | TPNET Polish Telecom_s commercial IP network
5617    | 217.98.63.167     | PL | TPNET Polish Telecom_s commercial IP network
5617    | 217.98.63.171     | PL | TPNET Polish Telecom_s commercial IP network
12476   | 62.121.117.72     | PL | ASTER-CITY-CABLE-AS Aster City Cable Sp. z o.o.
```

```
12824   | 62.129.253.44     | PL | HOMEPL-AS home.pl autonomous system
12741   | 62.233.128.22     | PL | INTERNETIA-AS Netia SA
5617    | 80.48.177.10      | PL | TPNET Polish Telecom_s commercial IP network
5617    | 80.50.50.10       | PL | TPNET Polish Telecom_s commercial IP network
5617    | 80.50.50.100      | PL | TPNET Polish Telecom_s commercial IP network
5617    | 80.53.164.186     | PL | TPNET Polish Telecom_s commercial IP network
5617    | 80.55.205.178     | PL | TPNET Polish Telecom_s commercial IP network
12741   | 81.219.165.18     | PL | INTERNETIA-AS Netia SA
30838   | 83.242.95.3       | PL | TELPOL PPMUE TELPOL
12968   | 85.128.40.3       | PL | CDP Crowley Data Poland, sp. z o.o.
```

10.4.3 Task 3 Looking further

You can tell that 10.16.54.2 is most likely a local mail server (many flows to port 25 TCP) and 10.16.54.6 is a web server (many flows to port 80 TCP) which was under DDoS attack. Apart from the obvious UDP flood, you can also spot some DNS requests (mostly to and from 10.16.54.29 which seems to be a local DNS server), ICMP traffic (not too interesting) and traffic from the local network.

Most of this can be easily spotted by using `grep -v` to filter out the lines we do not want to see from the output, eg:

```
$ grep –v UDP 24022007.txt
```

It may be worth checking what other traffic was hitting the web server and, considering the time of day, who was visiting the webpage.

```
grep 10.16.54.6 24022007.txt| grep -v UDP
```

As you can see, apart from the attack, there are regular TCP connections to the http server at port 80. Now we can see whether some are visiting more frequently than others…

```
$ grep 10.16.54.6 24022007.txt| grep -v UDP | awk '{print $5}'
| awk -F: '{print $1}' | sort
```

We limit the output to source IP addresses and sort them to see how many times they showed up. As you can see, two hosts stand out: 85.128.40.3 and 66.249.72.45 (10.16.54.6 comes from the lines with http server responses). We can take a closer look:

```
$ egrep "66.249.72.45|85.128.40.3" 24022007.txt
```

You can see that the hosts were visiting the website once a minute. The sample of the traffic is quite small (approx. 7 minutes) but normally during an attack this can be a potentially interesting track.

10.5 Summary of the exercise

Wrap up the exercise by comparing the methods used by different students. Are some of them more efficient than other? How could the same techniques be applied to different scenarios from the introductory discussion? What other tools may be needed?

10.6 EVALUATION METRICS

You can use the expected results and solutions provided above to evaluate the exercise. Keep in mind that there are a variety of ways to complete the exercise and students should chose the ones with which they feel most comfortable.