Security in Mobile Systems - UE/EX

FH Hagenberg, WS 2017
Dipl.-Ing. Dr. Erik Sonnleitner, OSCP

Exercise 4: Security Tools (1)

Exercise 4: Basic security testing (1)

(1) Use any netcat flavor to perform the following tasks:
- (1.1) Create a bind-shell on the host netcat is in server (listening) mode. A bind-shell allows netcat clients to use a server-side shell (execute commands on the server).
- (1.2) Create a reverse-shell. A reverse shell is the counterpart of a bind-shell, and allows the server (!) to execute commands on the client.
- (1.3) Can these shells be used when server and client are using different OSs?
- (1.4) Can (1.1) and/or (1.2) used with UDP?

(2) Construct server and client netcat command-lines which resemble simple a text-chat.
- Chat must be accessible to multiple clients simultaneously
- All clients must receive all messages from all other clients
- All traffic must be encrypted at all times
- Chat server must only allow predefined IPs
- Server must keep a central chat log

(3) What does the following command-line do?

```
while $(ncat -lp 8080 -c 'ncat localhost 80'); do true; done
```

Exercise 4: Basic security testing (1)

(4) Due to security awareness, some netcat versions prevent using the `-e` option (route input data to process stdin). However, there is a work-around for establishing bind-shells nevertheless (server-side):

```
mkfifo /tmp/f
cat /tmp/f | /bin/sh -i 2>&1 | ncat -lp 1234 > /tmp/f
```

Deconstruct this command and explain exactly what each element is doing.

(5) Use `ncat` to fetch your e-mails from your FH mail account via IMAP. See **`https://en.wikipedia.org/wiki/Internet_Message_Access_Protocol`** for how the IMAP protocol works.

(6) Use the `ncat` and `tar` commands to copy the entire contents of your `/etc/` directory from your computer to one of your colleagues. The file transfer must be encrypted and compressed. The compression must be done on-the-fly and must not consume additional disk space on the sending client. The receiver must decompress on-the-fly to a predefined sub-directory in his/her home directory. Both sender and client must only issue one single command-line (Enter key only pressed once).

Exercise 4: Basic security testing (1)

(7) Create a very simple shell-script, which iterates over URLs of web-servers in a given text-file (one per line), and use `ncat` to grab the webserver's banner (Hint: use the HTTP HEAD command). Grab the `Server:` line from each response, and store them in a dedicated text-file on your system.

URL file contents:

```
---begin---
orf.at
derstandard.at
diepresse.com
heise.de
golem.de
blog.fefe.de
bild.de
---end---
```

Which webserver software is used most often, and on what sites? Which ones are used least often?

Exercise 4: Basic security testing (1)

(8) By default, nmap only scans the 1000 mostly used ports for performance, and partly for stealth reasons. In order to force scanning all 64k TCP ports, we can use the following command-line:
`nmap -p- <target>`
Using the Linux firewall `iptables`, we can monitor and count TCP packets. In order to do this, we need to add two firewall rules (for inbound and outbound traffic), and reset the counter:
`iptables -I INPUT 1 --source <src ip> -j ACCEPT`
`iptables -I OUTPUT 1 --destination <dst ip> -j ACCEPT`
`iptables -Z` (reset counter)
To show network statistics, just view the iptables:
`iptables -vn -L`
Scan all TCP ports of one of your (or a colleague's) hosts, and find out (8.1) how many packets have been sent, and (8.2) how many traffic has been generated (bytes).

Exercise 4: Basic security testing (1)

(9) Nmap comes with a neat scripting engine (NSE) - installed scripts are found on most Linux distros in `/usr/share/nmap/scripts/` (see **http://nmap.org/nsedoc** for more details).
One of these scripts is `http-virustotal`, which is capable of utilizing the services of **http://www.virustotal.com**. A private API key is needed to use it (issued after registration).
Read the NSE documentation of this script, and write a small shell script which:

- Takes a file to scan as argument: `./avscan.sh <path-to-file>`
- From the given file, calculate a hashsum (MD5, SHA1 or SHA256)
- Invoke nmap to query Virustotal
- The only output should be the total count of anti-virus engines which do have recognized the given file as virus, and the count of those which didn't.
- Only one request to VirusTotal is allowed per script call
- For testing purposes, use the EICAR test file. See **https://en.wikipedia.org/wiki/EICAR_test_file** for more information