TECHNISCHE UNIVERSITÄT WIEN
Vienna University of Technology

institute of telecommunications

Exercise 5: A Backdoor Using a Covert Channel

Network Security - Advanced Topics (VU 389.160), Winter Semester 2018/2019

Communication Networks Group at the Institute of Telecommunications

Thanks to your efforts in analyzing past traffic and in decoding hidden messages, the Ministry has been able to identify the information leak and therefore took measures to limit the damage as well as to prevent future intrusions. The question where data has been leaked to is still unanswered, however.

Meanwhile, the security staff has taken the second suspect into custody and, after a short interrogation, the suspect has provided information on how data was transferred to some remote machine. By using a covert channel, the suspects were able to circumvent all security measures in place. The covert channel was used to execute commands on some remote machine and convey information in this way. Furthermore, the suspect has even provided information on how to access the screen of the remote machine. Unfortunately, he fainted then (mysteriously) ...

Nevertheless, the Ministry needs you for a last mission: find out whether the suspect's covert channel actually works and get information about the remote machine and the network to which the data was presumably leaked!

Information sent through covert channels can be used, for example, by backdoors and botnets to execute commands in vulnerable machines. For this exercise, a simple backdoor program listens to incoming IP datagrams on the remote machine. The IP address of the remote machine is $198.19.19.$*your team number*; i.e. if your team number is 42, then the IP address of the remote machine is $198.19.19.42$. The backdoor listens on TCP port 1337 and considers the *acknowledge sequence value* (ack-seq) as an ASCII character (see Figure 1). The backdoor actually implements a simple *remote command execution*; a *[Backspace]* character will delete the previous character sent through the covert channel and the *[CARRIAGE RETURN]* will execute the command formed by the previously sent characters after concatenating them.[1]

Visual access to the remote machine's screen allows you to check your progress while solving the exercise. On that screen you see a *terminal window* that provides feedback from the backdoor, and another terminal showing the output of tshark (the command line version of Wireshark). tshark shows the time, source IP, destination IP, source port, destination port, acknowledgement number, and sequence number. The output of the screen can be accessed by connecting to $198.18.0.1$ on port $3000+$*your team number*; i.e. if your team number is 23 than the port is 3023. Since you got visual access only, you can neither use the keyboard nor the mouse.

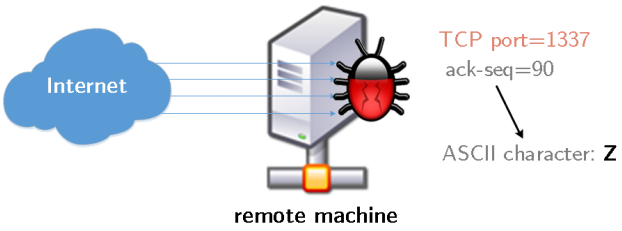

Figure 1. Backdoor listening to traffic.

Rep:5.a

First, establish a connection using *ssh* (username *nsa*) to the remote screen. Then, send a few pings to the remote machine and write in your report what you see on the remote screen, i.e.:

- The command(s) you used.
- The consequences of your connection attempts that you see in the remote screen (including tshark/Wireshark captures).

Answer the following questions in the report:

- Have you been able to reach the remote machine? Why (not)? Provide proper explanations.
- In case there is a firewall blocking the connections, is it a network layer firewall or an application layer firewall? Is it located on the remote machine or could it be in an intermediate router? Justify your answers.

[1] This exercise is loosely based on an article published by Security Art Work in: `http://www.securityartwork.es/2012/11/27/covert-channels-2/?lang=en`

5 Covert channels through intermediate servers

After all, it looks like some kind of firewall prevents sending commands directly to the backdoor on the remote machine, at least from your network. In order to circumvent that firewall, establish a covert channel through innocent web servers.

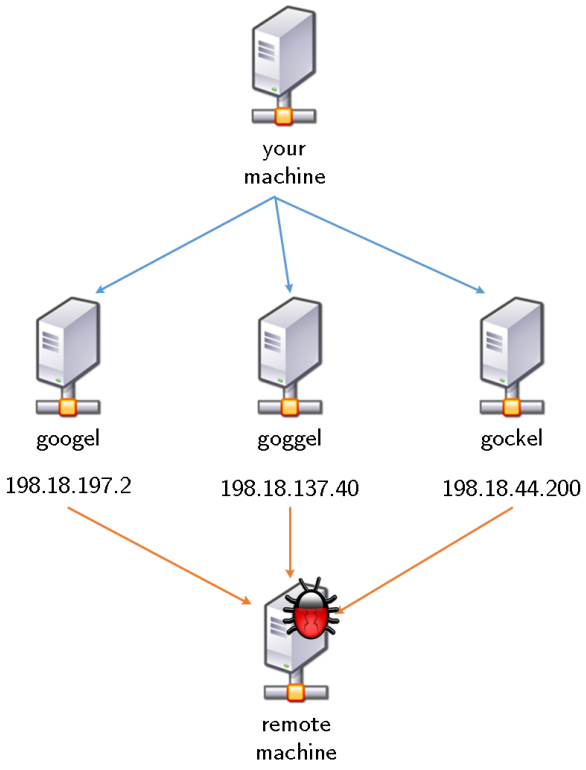

Figure 2. Covert channel through unaware servers.

The idea behind the scheme sketched in Figure 2 is to send SYN packets to external, non-related servers as if you were to start a TCP connection. However, instead of using your real IP address, the source IP field of packets is forged (as the address of the remote machine). Servers answer a SYN packet with the corresponding SYN-ACK to the IP address specified in the (faked) source field. In this way, information can be concealed. Some botnets and backdoors use similar, non-direct communication to make covert channels more difficult to detect and to hide the real source of the communication.

In order to avoid actions that could be considered illegal or malicious by third parties, use the servers in the lab provided for this exercise. **Do not send spoofed packets to external networks!**[2]

5.1 Generating arbitrary packets

For this exercise, you need to generate TCP/IP packets including header fields with arbitrary content. To this end, you can use the command line program c or the (more sophisticated) Python library Scapy, for example. Examples for `tgn` are provided in its mainpage. If you use Scapy, you do not need root privileges in the lab environment. Here is a short example for Scapy to get you started. For more information, read the online documentation `http://www.secdev.org/projects/scapy/doc/index.html`.

```
1 #!/usr/bin/env python
2
3 from scapy.all import *
4
5 send(IP(dst="127.0.0.1")/TCP(sport=31337))
```

Rep:5.b

Use the covert channel through the intermediate servers to get the network configuration of the remote machine (*ifconfig* or *ip addr*).

Write in the report the values of the header fields for every packet that you send in order to fulfill this task (required fields: source IP, destination IP, source Port, destination port, TTL, IP ID, sequence value, acknowledge sequence value). Provide explanations that justify the reported values, if they can vary or must be specific in order the command to succeed.

Note: On the remote machine you will see a notification explaining that the command has been executed, yet the output will not be displayed. To see the output you have to send it back to your machine, using *netcat* for example (see next subsection).

5.2 Netcat (nc) – a swiss army knife for TCP/IP

To accomplish your last mission, get information about the remote machine and the network. To send the output of a command back to your computer `netcat` can be used. It allows sending data across networks using TCP or UDP from the command line. Examples are provided in netcat's manpage.

Rep:5.c

Create a script that performs the whole task, i.e. (1) connect from your machine to the backdoor using a covert channel through different intermediate servers to overcome the security system, (2) run the commands that send the network configuration of the remote machine to your computer.

Add your script to the report!

[2] The authors of this document generally do not approve illegal activities, but we do like spy movies!