Network 101 - training

Maël Auzias

Fall 2015

Contents

1 HTTP example

1.1 Who are you? Where are you?

What is your own IP address? What is your MAC address? What is your network mask? Do you have an IPv6 address? What do theses commands display?

```
$ifconfig
$curl https://icanhazip.com
$netstat -at
$nmap localhost
$ping 3564020356.org
```

1.1.1 How to get out?

Before we can access the Internet we need to know who/what is the gateway. What is a gateway? What do theses commands display?

```
#route -n
#arp -a
```

1.1.2 What's your number?

As explained before, humans can easily remember names such as news.ycombinator.comor root-me.orgbut it is not as easy to remember `198.41.191.47` or `212.129.28.16`. We need a way to *translate* a domain name into an IP address. This is role of DNS[1]. You can query DNS using `nslookup`.

1.1.3 Wait! What direction?

The IP address of the website you want to visit is now known. The next step is to know how to get there. Try to trace the route using `traceroute` (or `tracepath`) to see packets' path. Do you know any town on the path from where you are to `www.ethicalhacker.net` server? Note that sometimes, for security reasons, ICMP protocol is blocked. If this is the case you can use an option to use TCP SYN for probes. How does `traceroute` work ?

1.1.4 Go GET it!

What does `wget 149.154.167.119 80` do?

1.1.5 Capture it

Use `wireshark` to capture:

- a GET through HTTP (selfoss.aditu.dedoes not have valid HTTPS certificate).
- a GET through HTTPS (micahflee.comforce redirection to HTTPS).

What differences can you see? How can you explain theses differences?

[1]DNS: Domain Name Server, if you needed to read this footnote keep in mind that you should remember it from now on

1.1.6 "Security" *without* HTTPS

Some methods allow web-master to secure some part of the website. Then the website requires a user and a password to enter. You can test on the webpage: http://teaching.auzias.net/http-auth/

- user: test
- pass: p4ssw0rd

Use `wireshark` and verify if you had captured the user:password. Is it encrypted? A fast reading of RFC 2617could be helpful.

2 Chat

2.1 netcat

`netcat (or ncat)` is a "network swiss army knife". By checking its man page how can you use it as a chat server/client (two nodes only).

2.1.1 TCP

Use the mode TCP of `netcat` and try it. Can `netstat`, somehow, be helpful for anything while waiting for connection? Can `telnet` be used to chat?

2.1.2 UDP

Use the mode UDP of `netcat` and try it. Explain a situation where the server could not receive every packet.
More example of netcat:brianhaddock.com

2.2 Create your own client

In the next two exercises you can use `netcat` to verify, whether or not, your client is working.

2.2.1 TCP

Create a TCP client that send packet to a specific port on `localhost`. (The class `Socket` should be useful...)

2.2.2 UDP

Try to produce the situation explained in 2.1.2 by implementing it using Java language and flood a specific port on `localhost`. (The classes `InetAddress` and `DatagramPacket` should be useful...)

2.3 Create your own server

In the two next exercise you can use your previous client to verify, whether or not, your server is working.

2.3.1 TCP

Implement in Java a TCP server that connects at the first attempt of connection and displays it on the screen. Next step is to *echo* it (send it back).

2.3.2 UDP

Implement in Java a UDP server that receives and displays it on the screen. Next step is to *echo* it (send it back).