Computer Networks Notes

Lessons Learned

Ladislav Šulák <laco.sulak@gmail.com>

April 28, 2021

Contents

1 General **2**
- 1.1 Internet Connection Troubleshooting . . . 2
- 1.2 Local Network Troubleshooting . . . 3
- 1.3 WebSockets . . . 3

2 Pentesting Tools **6**
- 2.1 Eavesdropping . . . 6
- 2.2 Packet Generation . . . 6
- 2.3 Testing . . . 7
- 2.4 Scanning . . . 8
- 2.5 Fingerprinting . . . 8
- 2.6 Attacks . . . 8
- 2.7 IDS / IPS . . . 9
- 2.8 Major firewall platforms . . . 9

3 Network Attacks **10**
- 3.1 Communication in local IPv4 network . . . 10

4 References **12**

1 General

- **Bandwidth**: This is the maximum amount of data that can be transferred in a unit of time. It is typically expressed in bits per second (or some similar ways, such as gigabytes per second).
- **Throughput**: Whereas bandwidth is the maximum data that can be transferred in a unit of time, throughput is the actual amount of data that is transferred. Bandwidth is the number of items that can be transferred per unit of time, in the best possible conditions. Throughput is the time it really takes, when the machines perhaps aren't operating smoothly.
- **Latency**: This is how long it takes data to go from one end to the other. That is, it is the delay between the sender sending information (even a very small chunk of data) and the receiver receiving it.

1.1 Internet Connection Troubleshooting

Step by Step:

1. **L3 - is an IP address allocated correctly?**

   Windows:

   - ipconfig /?
   - ipconfig /all
   - ipconfig /release
   - ipconfig /renew

   Linux:

   - *ip address*
   - *ip route*

2. **Mapping L3 and L2 - is it working properly?**
   - *arp*
   - *ip neigh*
   - *netsh interface ipv6 show neighbors*
3. **Switching - is correctly set?** What's the default gate?

- IP, route, ipconfig
- test ping
- traceroute/tracert

4. **Is DNS working properly?** What is your DNS resolver, can I use it and translate a domain name?
   - ipconfig /all
   - cat /etc/resolv.conf
   - ping $SERVER_IP
   - telnet $SERVER_IP 53
   - *dig | nslookup | host seznam.cz*

1.2 Local Network Troubleshooting

- **L1** - kabelaz, pomocou testeru, overenie linky medzi uzivatelom a prepinacom, ci su trasy a kable spravne zapojene
- **Topology**
- **Services-** monitorovanie dostupnosti (nie len dostupnost ale aj zatazenie atd) sluzieb (DHCP, DNS) a monitorovanie zarizenich (SNMP)
- **Security** - DHCP snooping, port security, ACL, ARP snooping
- **Others** - Netflow - statistika, uctovanie, dohladavanie problemov a dotazovanie

1.3 WebSockets

- Communication protocol, provides full-duplex communication channels over a single TCP connection. Standardized in 2011.
- WebSocket is distinct from HTTP. Both protocols are located at layer 7 in the OSI model and depend on TCP at layer 4. Websocket has also handshake. However[1]:
  - Persistent stateful connection for the duration of connection
  - Low latency: near real-time communication between server/client due to no overhead of reestablishing connections for each request as HTTP requires.
  - Full duplex: both server and client can send/receive simultaneously
  - WebSocket and HTTP protocol have been designed to solve different problems, I.E. WebSocket was designed to improve bi-directional communication whereas HTTP was designed to be stateless, distributed using a request/response model.

[1] https://stackoverflow.com/questions/14703627/websockets-protocol-vs-http

  - The basic thing behind both the WebSocket and HTTP is the socket. In HTTP, it opens a connection on request and closes on response. For WebSocket, concept is a 2 way communication (full duplex) rather than request-response cycle.

- Websockets vs RESTful:[2]
  - With RESTful HTTP you have a stateless request/response system where the client sends request and server returns the response.
  - With WebSockets you have a stateful (or potentially stateful) message passing system where messages can be sent either way and sending any given message is lower overhead than with a RESTful HTTP request/response.

- The WebSocket protocol enables interaction between a web browser (or other client application) and a web server with lower overhead than half-duplex alternatives such as HTTP polling, facilitating real-time data transfer from and to the server.

- In WebSocket, it's using ping-pong mechanism to make sure that the client or the server is alive. For every ping requests from one end, other end is subjected to reply a pong response. This mechanism helps to detect failures and hence to maintain stability.

- **Advantages of WebSockets over RESTful**
  - Two way communication. So, the server can notify the client of anything at any time. So, instead of polling a server on some regular interval to see if there is something new, a client can establish a WebSocket and just listen for any messages coming from the server. From the server's point of view, when an event of interest to a client occurs, the server simply sends a message to the client. The server cannot do this with plain HTTP.
  - Lower overhead per message than RESTful HTTP. This is because the TCP connection is already established and you just have to send a message on an already open socket. With an HTTP REST request, you have to first establish a TCP connection which is several back and forth between client and server. Then, you send HTTP request, receive the response and close the TCP connection. The HTTP request will necessarily include some overhead such as all cookies that are aligned with that server even if those are not relevant to the particular request.
  - Stateful connections. Without resorting to cookies and session IDs, you can directly store state in your program for a given connection. While a lot of development has been done with stateless connections to solve most problems, sometimes it's just simpler with stateful connections.

[2] https://stackoverflow.com/questions/29925955/what-are-the-pitfalls-of-using-websockets-in-place-of-restful-http

- **Advantages of RESTful over WebSockets**
  - Universal support.
  - Compatible with more server environments.
  - For a one-off request/response, a single HTTP request is more efficient than establishing a WebSocket, using it and then closing it. This is because opening a WebSocket starts with an HTTP request/response and then after both sides have agreed to upgrade to a WebSocket connection, the actual WebSocket message can be sent.
  - Stateless. If your job is not made more complicated by having a stateless infrastructure, then a stateless world can make scaling much easier.
  - Automatically cacheable. With the right server settings, http responses can be cached by browser or by proxies. There is no such built-in mechanism for requests sent via WebSockets.
- An alternative: WebRTC[3]:
  - API definition drafted by W3C that supports browser-to-browser communication without plugins.
  - It allows audio and video communication to work inside web pages by allowing direct peer-to-peer communication, eliminating the need to install plugins or download native apps.

[3] https://stackoverflow.com/questions/14703627/websockets-protocol-vs-http

2 Pentesting Tools

2.1 Eavesdropping

- *net2pcap*
- *cdpsniffer*
- *aimsniffer*
- *vornst*
- *tcptrace*
- *tcptrack*
- *nstreams*
- *argus*
- *karpski*
- *ipgrab*
- *nast*
- *cdpr*
- *aldebaran*
- *dnsniff*
- *apas*
- *iptraf*

2.2 Packet Generation

- *scapy (the most common one)*
- *packeth*
- *packet*
- *packet exkalibur*

- *nemesis*
- *libnet IPsorcery*
- *pacgen*
- *arp.sk*
- *arpspoof*
- *dnet*
- *dpkt*
- *iripas*
- *sendIP*
- *IP-packetgenerator*
- *sing*
- *libpal*
- *aicmpsend*

2.3 Testing

- ping
- *hping2*
- *hping3*
- *traceroute*
- *mtr (! pinguje zistuje cestu a stabilitu a odozvu. V podstate ako traceroute, ale bezi stale a zistuje aj stratu paketov atd.)*
- *tctrace*
- *tcptraceroute*
- *traceproto*
- *tping*
- *arping*

2.4 Scanning

- *nmap*
- *amap*
- *vmap*
- *hping3*
- *unicomscan*
- *ttlscan*
- *paketto*
- *firewalk*

2.5 Fingerprinting

- *nmap*
- *xprobe*
- *p0f*
- *cron-OS*
- *queso*
- *amap*
- *syscan*

2.6 Attacks

- *dnssproof*
- *poison ivy*
- *Macof*
- *ikeprobe*
- *ettercap*
- *dnsniff suite*
- *cain*

- *hunt*
- *airpwn*
- *irpass*
- *nast*

2.7 IDS / IPS

- Open-source: *Snort, Bro, Surricata*
- Enterprise: *Fortigate, Cyberoam, Watchguard*
- A simple IDS/IPS system that logs /var/log/auth.log - *Fail2ban*

2.8 Major firewall platforms

- *Checkpoint*
- *Juniper*
- *Cisco*
- *Fortinet*

3 Network Attacks

3.1 Communication in local IPv4 network

- *Access to a website*
  1. Pridelenie adresy pomocou DHCPv4 - **DHCP spoofing** (typicky nie je nic podpisovane ani sifrovane) - poslem ako klient do sieti 'DHCP discover' a kedze je to broadcast tak sa rozposle na vsetky prvky. Niektore spravu zahodia a DHCP server jediny vygeneruje 'DHCP offer' s5 ku mne. Ja poslem nasledne 'DHCP request' (posle sa opat broadcastovo lebo moze existovat viac DHCP serverov). Server posle 'DHCP acknowledge' ze spravu potvrdzuje a klient si moze u seba nastavit IP adresu. Az potom si ju klient nastavi u seba.
  2. ziskanie adresy MAC brany (**CAM overflow**, alebo **utoky na ARP protokol**).
  3. mapovanie domenoveho mena -> IP (pomocou DNS).
  4. naviazanie spojenia pomocou TCP a nasledne HTTP request.
- *DHCP Spoofing*
  - Kedze sa sprava na zaciatku rozposle vsektym, aj utocnikovi. Jednym zo sposobov ako klientovi spravu podvrhnut, je ten, ze sa ju (DHCP offer) **skusim dorucit skor** (zahltit ho aby bol pomalsi) **nez DHCP server**. Nasledne si nastavim informacie podla toho, co mi dal utocnik. Utocnik napriklad da vychodzu MAC branu na seba, ze je to on sam. Teda vsetok prenos klienta do internetu pojde cez neho.
  - **Zneuzitie** - ukradnutie IP adresy (ziskanie napr pristupovych prav niekde inde), presmerovanie vychodzej brany, vlozenie vlastneho DNS serveru, falosny DHCP server - trojany.
  - **Realizacia** - ettercap for MITM.
  - **Obrana** - switch bude filtrovat tieto spravy, teda switch bude povolovat iba spravy z DHCP serveru. Monitovaci system (IDS/IPS). Switch bude mat povoleny port na DHCP offer iba od praveho DHCP serveru. Problem je ten, ze switch pracuje na L2 a aby sa dopracoval az do aplikacneho protokolu aby zistil ze sa jedna o DHCP offer stoji vykon. Mozeme skusit filtrovat pomocou IP adresy DHCP serveru napriklad.
- *CAM Overflow*

  - zaplnenie CAM tabulky - uklada info o porte, VLAN, MAC a stari zaznamu. Zaznamy z nej nemozu byt vytlacene. Utok funguje iba po dobu vyprsania zaznamu z tabulky a sila je iba na sniffing. Vysledkom je to, ze mozem ako utocnik odposluchavat provoz, ku ktoremu by som sa normalne nedostal.
  - **Realizacia** - Macof
  - **Obrana** - Port security - povolene iba predkonfigurovane MAC adresy, obmedzit pocet MAC adries na port. Dopad to ma taky, ze mame iba obmedzeny pocet zariadeni a tak istym sposobom limitujeme uzivatela. Casto je to implementovane v HW a nema to dopad na vykon.

- *ARP Spoofing*
  - Pokusim sa z pohladu utocnika preucit pocitace, aby isla komunikacia cezo mna (MitM zas). Na zaklade toho ze ku kazdej IP adrese je nejaka MAC adresa, otrava cache.
  - **Obrana** - Dynamic ARP Inspection - tvorba tabulky MAC-IP-Port, testovanie ARP paketov a povolenie iba spravnych. Switch aj tak spracovava tieto informacie (**DHCP snooping**), no i tak to ma jednoznacne dopad moc vykon. Dalsi dopad - zamedzenie statickych IP adries, ak sa nejaka potrebna vazebni informacia nenachadza (DHCP binding), spravu zahodim. Okrem toho restart zariadenia znemozni klientom komunikaciu.

4 References