Vysoké Učení Technické v Brně

Fakulta informačních Technologií

Počítačové Komunikace a sítě

2019/2020

Dokumentace k projektu IPK

Zadání OMEGA - Scanner síťových služeb

Kryštof Halmo (xhalmo00) Brno, 3.5.2020

Obsah

1. Úvod

Tento projekt byl zadán jako úkol do předmětu IPK, kde studenti měli naimplementovat scanner síťových služeb, který má na dané adrese skenovat TCP a UDP porty. Tento prvek slouží k odhalení zranitelnosti server, která může vést k jeho opravě nebo jeho zneužití.

1.1 Nástroje pro testování

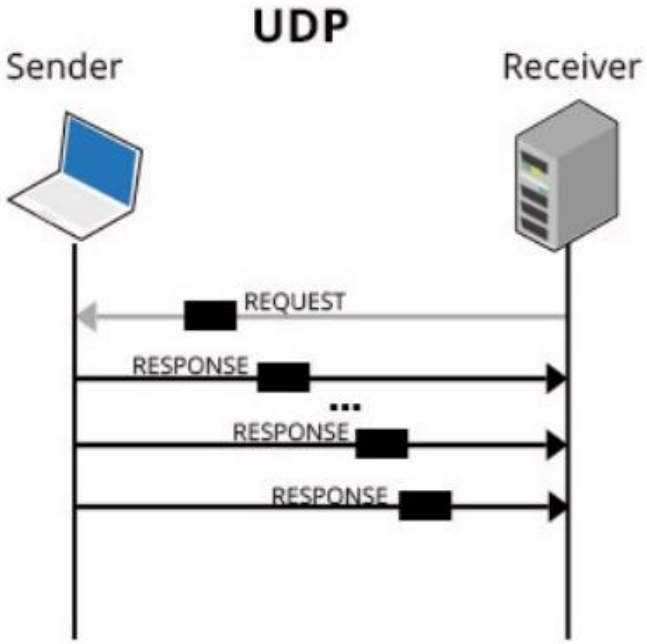
Pro ověření validity výstupu projektu byl použit nástroj wireshark[1] a webové stránky, které dodávají informace o portech na zadaných IP adresách[2].

2. Protokoly

Protokoly, které byly použity jsou IP, TCP, UDP[3].

2.1 UDP Protokol

UDP (User Datagram Protocol) je protokol založený na “Best effort delivery”, což znamená, že packet se pokusí doručit, ale jeho doručení nezaručuje. Zároveň je “Connectionless”, což znamená, že se nevytváří spojení se serverem. Od serveru nedochází potvrzující packet, tedy nedochází k takzvanému potřesení rukou v případě našeho projektu, pošleme udp dotaz na zadanou IP adresu a zadaný port. Pokud neobdržíme odpověď nebo obdržíme ICMP packet, tak je port zavřený. V případě, že odpověď obdržíme, tak je port otevřený[4].



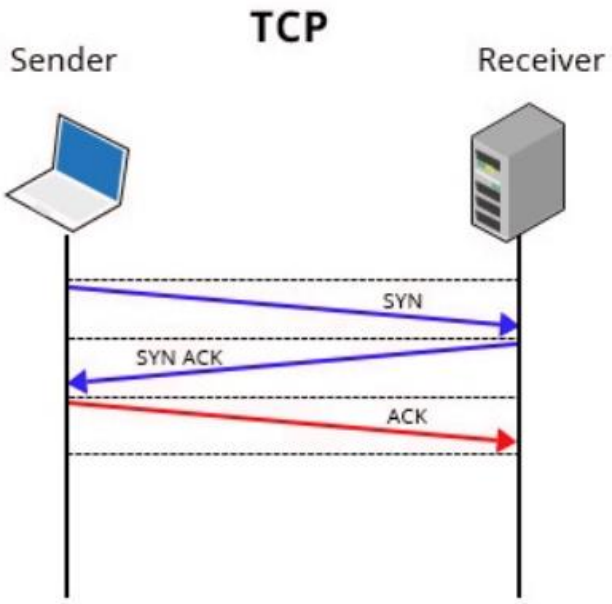
Obrázek 1: UDP protokol 1

2.2 TCP Protocol

TCP (Transmission Control Protocol) je narozdíl od UPD na principu “Guaranteed delivery”. Zaručuje nám doručení paketu, a to tak, že každou žádost je posláno potvrzení o doručení. Paketem se posílá i kolik chceme zaslat dat a pořadí paketů, tedy když server paket neodrží, pak poprosí klienta o znovuzaslání.

Také je to “Connection-oriented protocol“, navazuje tedy spojení pomocí tzv. “Three-way handshake”, kde klient pošle serveru paket, ve kterém jej požádá o komunikaci, na což server pošle odpověď, kde specifikuje, jak server může komunikovat. Klient si vybere jednu z těchto možností a pošle odpověď se specifikovanými údaji.

Pro naši implementaci to znamená, že pokud obdržíme odpověď od server, tak je buď port otevřený nebo zavřený, a to na základě typu odpovědi. Pokud nám server neodpoví, tak je tento port je blokovaný aneb filtrovaný[5].



Obrázek 2: TCP protokol

3. Implementace

Projekt je implementovaný v jazyce C++.

3.1 Argumenty

Pro získání argumentu jsou implementované funkce:

- ProcesArgs: tato funkce zjistí, jestli jsou všechny argumenty zadané správně a pokud nejsou zadané a nejsou povinné, tak zavolá funkce, které je doplní. Argumenty jsou testovány pomocí funkce getopt_long[6].

- getPortArray: je funkce, která převede zadané porty ze vstupního formátu na zásobník čísel.
- ValidIpAddress: otestuje, jestli je IP adresa platná, na základě počtu teček a rozsahu mezi jednotlivými tečkami a vrátí odpověď.
- ValidDomain: otestuje, jestli se jedná o doménové jméno na základě teček a jestli se ve stupni proměnné nachází www a vrátí odpověď.
- getIpAdressFromName: pokusí se převést doménové jméno na IP adresu a vrátí odpověď nebo ukončí program. Pomocná funkce gethostbyname.[7]
- getInterface: je funkce, která je volána jen pokud není zadané rozhraní. Najde první neloopbackové rozhraní a vrátí jeho název.[8]
- getIpAddress: je funkce, která se pokusí zjistit IP adresu k rozhraní a vrátí ji.[9]
- PrintHelp: je funkce, co vypíše pomocný výpis.

3.2 UDP Skener

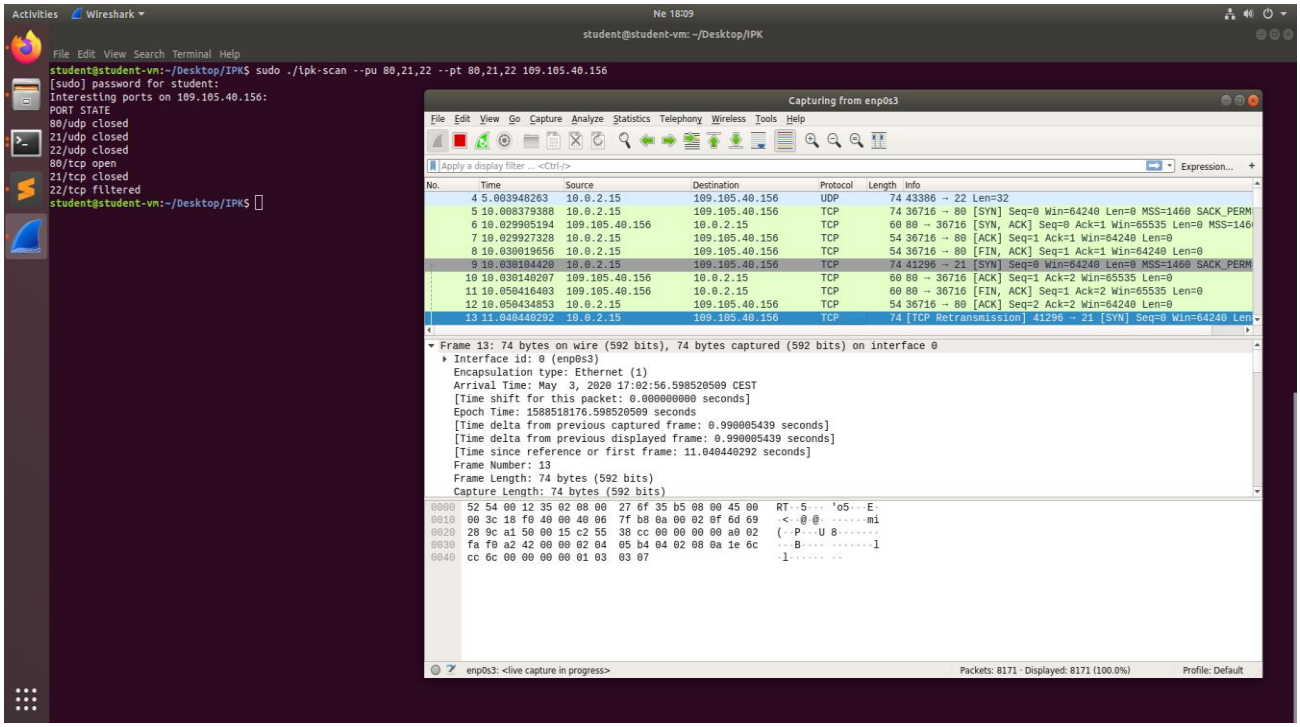
Je funkce, ve které jsou deklarovány a naplněny UDP a IP struktury. V této funkci je vytvořen socket, který je dále modifikován, aby přes něj bylo možné odesílat UPD packet. Tento packet je pak poslán pomocí funkce sendto a je nastaven odpočet, aby nebylo možné se zaseknout při čekání na odpověď.[10] Dále následuje čekání a odpověď funkcí recvfrom. Na základě výsledků funkce recvfrom je vypsán stav portu.[11]

3.3 TCP Skener

Je funkce, ve které jsou deklarovány a naplněny TCP a IP struktury. Je vytvořen socket, který je dále modifikován, aby přes něj bylo možné odesílat TCP packet. Před navázáním spojení se nastaví časovač, aby nebylo možné se zaseknout při čekání na odpověď.[12] Spojení je navázané pomocí funkce connect a základě výsledku je vypsán stav proudu.[13]

4. Testování

Testování proběhlo za pomocí programu wireshark[1], na kterém bylo testováno, zda se packety vůbec posílají a jestli chodí odpovědi.



6. Zdroje

[1] https://www.wireshark.org
[2] https://www.ipfingerprints.com/portscan.php) (https://dnslytics.com/ip/8.8.8.8

[3] IP: https://tools.ietf.org/html/rfc768,

TCP: https://tools.ietf.org/html/rfc793,

UDP: https://tools.ietf.org/html/rfc791

[4] https://wis.fit.vutbr.cz/FIT/st/cfs.php?file=%2Fcourse%2FIPK-IT%2Flectures&cid=13334

[5] https://wis.fit.vutbr.cz/FIT/st/cfs.php?file=%2Fcourse%2FIPK-IT%2Flectures&cid=13334

[6] https://www.gnu.org/software/libc/manual/html_node/Getopt-Long-Option-Example.html

[7] https://paulschreiber.com/blog/2005/10/28/simple-gethostbyname-example/

[8] http://man7.org/linux/man-pages/man3/getifaddrs.3.html

[9]https://gist.github.com/quietcricket/2521037?fbclid=IwAR0qv9CPWL_Zw575aweSQIqzwTTI5Dqz4oaAcygqKg2xB4oM44xvsxU5ptc

[10] http://www.mathcs.emory.edu/~cheung/Courses/455/Syllabus/9-netw-prog/timeout.html

[11]https://www.tenouk.com/Module43a.html?fbclid=IwAR3J6iPdapuYLLIGgNScM1k0Nh3LwFj6hQ01Xr87fBd8c-EDRdg2492wKIU

[12] https://stackoverflow.com/questions/4181784/how-to-set-socket-timeout-in-c-when-making-multiple-connections?fbclid=IwAR1dEY1MVG6YBAo-HYEP8bjOUHku45YN5IioVsWBgcZMhkEm3bxQPy7do1s

[13] https://docs.python.org/2/library/errno.html

https://www.tenouk.com/Module43a.html?fbclid=IwAR3J6iPdapuYLLIGgNScM1k0Nh3LwFj6hQ01Xr87fBd8c-EDRdg2492wKIU

6.1 Obrázky

Obrázek 1: UDP protokol: https://www.muvi.com/wiki/udpuser-datagram-protocol.html

Obrázek 2: TCP protokol: https:www.muvi.com/wiki/udpuser-datagram-protocol.html