

VYSOKÉ UČENÍ TECHNICKÉ V BRNĚ
FAKULTA INFORMAČNÍCH TECHNOLOGIÍ

IPK - Počítačové komunikace a sítě

Projekt 2 - Varianta DELTA: Scanner síťové
dostupnosti

1 Úvod do problematiky

Zisťovanie či je host aktívny pomocou ICMP echo requestu nemusí vždy vypovedať o tom, či je host naozaj aktívny. Väčšina dobre zabezpečených zariadení v sieti lokuje prijímanie týchto paketov, aby sa porty zbytočne nezahľcovali odpovedaním na tieto requesty. V lokálnej sieti sa dá s väčšou istotou použiť ARP request, kde sa ako odpoveď vracia MAC adresa zariadenia. Odpoveď na ARP request je rýchla, a ak sa neobjaví po krátkom čase dá sa s istotou povedať že zariadenie nie je aktívne.[1] V prípade že máme IPv6 adresu, používa sa NDP, ktorého funkcia je v podstate rovnaká ARP.

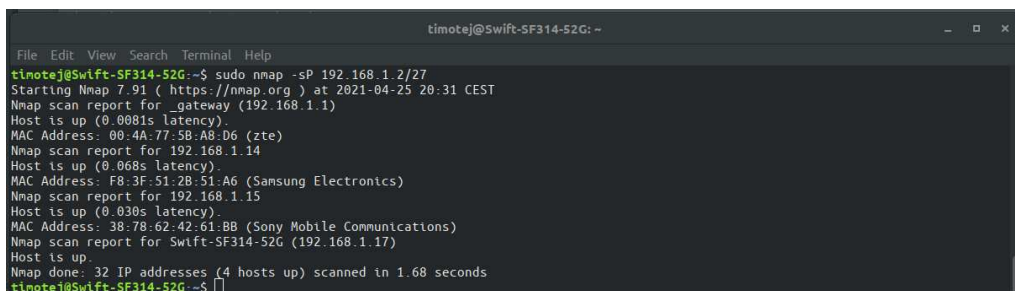
2 Implementácia

V mojej variante projektu išlo hlavne o rozpoznanie rozsahu skenovaných adries a postupné odosielanie ARP/ICMP paketu v prípade IPv4, NDP/ICMP v prípade IPv6, čakanie na odpoveď a výpis odpovede na stdout. Dôležité bolo pochopiť ako funguje CIDR[2] notácia adries, kde sa zo sufixu dá jednoducho vypočítať počet ip adries, zložitejšie to je s určením rozmedzia IP adries v subnete.

3 Testovanie

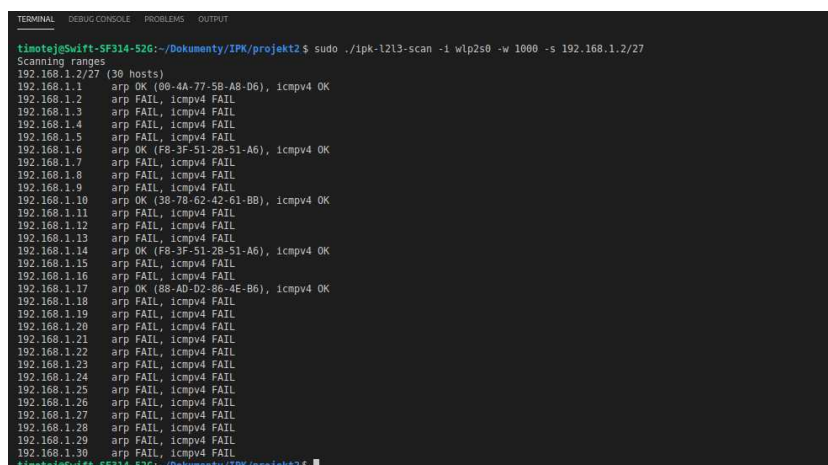
3.1 Testovanie skenovania

Výstup programu bol porovnávaný s výstupom programu nmap



```
timotej@Swift-SF314-52G: ~  
File Edit View Search Terminal Help  
timotej@Swift-SF314-52G:~$ sudo nmap -sP 192.168.1.2/27  
Starting Nmap 7.91 ( https://nmap.org ) at 2021-04-25 20:31 CEST  
Nmap scan report for _gateway (192.168.1.1)  
Host is up (0.0081s latency).  
MAC Address: 00:4A:77:5B:A8:D6 (zte)  
Nmap scan report for 192.168.1.14  
Host is up (0.068s latency).  
MAC Address: F8:3F:51:2B:51:A6 (Samsung Electronics)  
Nmap scan report for 192.168.1.15  
Host is up (0.030s latency).  
MAC Address: 38:78:62:42:61:BB (Sony Mobile Communications)  
Nmap scan report for Swift-SF314-52G (192.168.1.17)  
Host is up.  
Nmap done: 32 IP addresses (4 hosts up) scanned in 1.68 seconds  
timotej@Swift-SF314-52G:~$
```

Obr. 1: Výstup z nmap



```
timotej@Swift-SF314-52G:~/Dokumenty/IPK/projekt2$ sudo ./ipk-l2l3-scan -i wlp2s0 -w 1000 -s 192.168.1.2/27  
Scanning ranges  
192.168.1.2/27 (30 hosts)  
192.168.1.1 arp OK (08-4A-77-5B-A8-D6), icmpv4 OK  
192.168.1.2 arp FAIL, icmpv4 FAIL  
192.168.1.3 arp FAIL, icmpv4 FAIL  
192.168.1.4 arp FAIL, icmpv4 FAIL  
192.168.1.5 arp FAIL, icmpv4 FAIL  
192.168.1.6 arp OK (F8-3F-51-2B-51-A6), icmpv4 OK  
192.168.1.7 arp FAIL, icmpv4 FAIL  
192.168.1.8 arp FAIL, icmpv4 FAIL  
192.168.1.9 arp FAIL, icmpv4 FAIL  
192.168.1.10 arp OK (38-78-62-42-61-BB), icmpv4 OK  
192.168.1.11 arp FAIL, icmpv4 FAIL  
192.168.1.12 arp FAIL, icmpv4 FAIL  
192.168.1.13 arp FAIL, icmpv4 FAIL  
192.168.1.14 arp OK (F8-3F-51-2B-51-A6), icmpv4 OK  
192.168.1.15 arp FAIL, icmpv4 FAIL  
192.168.1.16 arp FAIL, icmpv4 FAIL  
192.168.1.17 arp OK (88-AB-D2-B6-4E-B6), icmpv4 OK  
192.168.1.18 arp FAIL, icmpv4 FAIL  
192.168.1.19 arp FAIL, icmpv4 FAIL  
192.168.1.20 arp FAIL, icmpv4 FAIL  
192.168.1.21 arp FAIL, icmpv4 FAIL  
192.168.1.22 arp FAIL, icmpv4 FAIL  
192.168.1.23 arp FAIL, icmpv4 FAIL  
192.168.1.24 arp FAIL, icmpv4 FAIL  
192.168.1.25 arp FAIL, icmpv4 FAIL  
192.168.1.26 arp FAIL, icmpv4 FAIL  
192.168.1.27 arp FAIL, icmpv4 FAIL  
192.168.1.28 arp FAIL, icmpv4 FAIL  
192.168.1.29 arp FAIL, icmpv4 FAIL  
192.168.1.30 arp FAIL, icmpv4 FAIL  
timotej@Swift-SF314-52G:~/Dokumenty/IPK/projekt2$
```

Obr. 2: Výstup z programu

Po porovnaní jednotlivých výstupov je vidieť že mnou vytvorený program negeneruje rovnaké výstupy ako nmap, ale pre skenované ip adresy získava správnu MAC adresu. Niekedy vypíše rovnakú mac adresu pre viacero IP adries (túto chybu je možné vidieť aj na screenshote), stáva sa to občas. V tomto prípade sa to možno stalo aj preto že som sa pokúsil spustiť nmap zároveň s mojím programom a socket ktorý odchyťava arp odpoveď odchytil odpoveď určenú pre nmap, ale stalo sa mi to aj pri bežnom testovaní.

3.2 Testovanie rozsahu IP adries

Na testovanie správneho počítania rozsahu adries bola použité online kalkulačky:

<https://www.vultr.com/resources/subnet-calculator-ipv6/>

<http://jodies.de/ipcalc> V kóde sa aj odkazujem na StackOverflow, kde som našiel kód ktorý správne ráta rozsah IPv4 adries pre zadaný subnet, tento kód som následne upravil pre získanie správneho rozsahu IPv6 adries, najdôležitejšie bolo si uvedomiť že IPv6 adresa má rozsah 128 bitov čo sa nedá reprezentovať datovým typom, takže sa muselo podľa prefixu bitovo posúvať cez pole. To bol hlavný rozdiel v počítaní rozsahu adries medzi IPv4 a IPv6.

4 Nedostatky a obmedzenia

Nie je implementované skenovanie IPv6 adries, v prípade zadania IPv6 subnetu sa iba vypíše rozsah ip adries, a skenovanie sa preskočí. Čakanie na odpoveď pre jednotlivé skenovanie IP adresy môže trvať dlhšie ako je nastavený parameter `wait`, nastavujem ho totiž zvlášť pre ARP a ICMP ping, zo zadania chápem že by to takto asi nemalo byť, ale nemal by to byť problém pretože kladná odpoveď na ARP request (MAC adresa), príde zväčša hneď, a ICMP echo posielam iba ak ARP získal MAC adresu, takže väčší čas čakania by sa nemal prejaviť.

5 Záver

Projekt ma bavil kým som si k nemu študoval materiály snažil sa pochopiť čo sa po mne chce a zbieral si informácie. Keď som sa dostal k implementácii tak sa môj postup zastavil, strávil som veľa času googlením ako získať rozsah ip adries zo subnetu, ako odoslať a prijať ARP paket a ICMP paket a podobne. Je to vidieť aj v kóde kde sa vo viac ako polovici kódu odkazujem na StackOverflow a podobné zdroje. Myslím si že v pythone by sa tento projekt písal ľahšie, ale chápem že sme si mali vyskúšať programovanie so soketmi v C/C++ alebo C#.

Literatura

[1] [online], [vid. 2021-04-25]. Dostupné z: <https://nmap.org/book/host-discovery-techniques.html#>

[2] [online], [vid. 2021-04-25]. Dostupné z: <https://whatismyipaddress.com/cidr>