Vysoké učení technikcé v Brně
Fakulta informačních technologií

Počítačové komunikace a sítě

Varianta OMEGA: Scanner síťových služeb

Peter Havan (xhavan00)

21. dubna 2019

Obsah

1 Zadanie

Úlohou v projekte bolo vypracovanie aplikácie na skenovanie TCP a UDP portov. Práca na projekte pozostávala z podúloh:

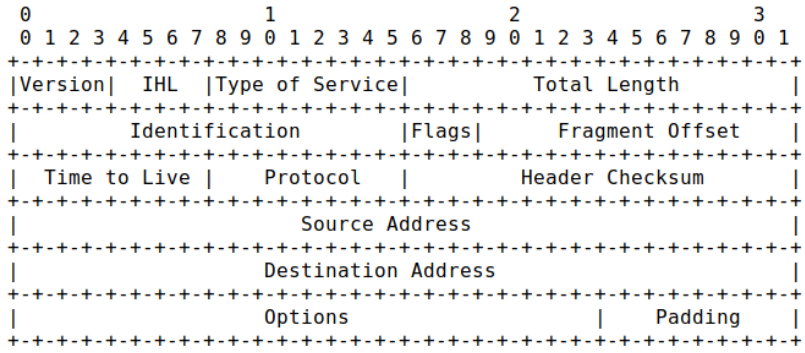
- Štúdium protokolov `IPv4, IPv6, TCP, UDP, ICMP`
- Implementácia `SYN` a `UDP` port scanneru v jazyku `C/C++` za použitia `BSD Sockets` a knižnice `libpcap`
- Vypracovanie dokumentácie/manuálu k projektu

2 Relevatné informácie

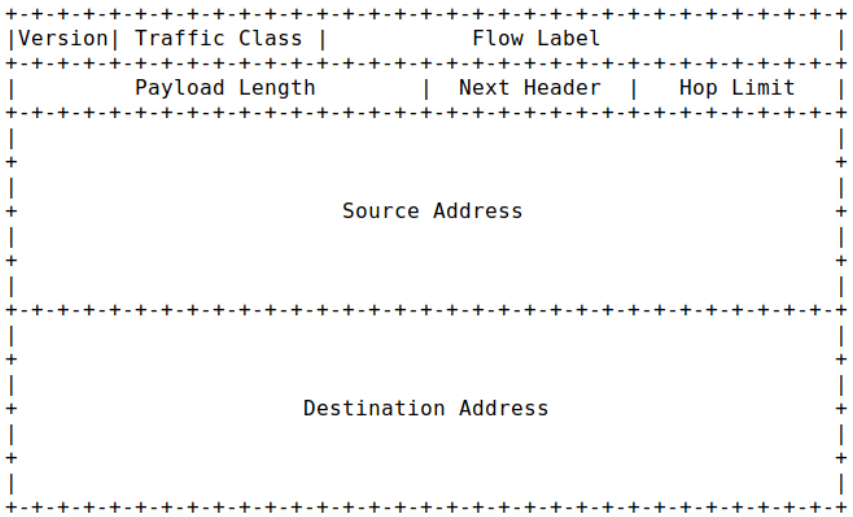
V tejto sekcií sa pozrieme na dôležitú teóriu, ktorá bola potrebná k vypracovaniu.

2.1 IPv4 a IPv6

Pre vytvorenie a odoslanie raw paketu je nutné vytvoriť a vyplniť IP hlavičku. Dôležité bolo štúdium jej formátu a najmä rozdiel medzi IPv4 a IPv6 hlavičkou.



Obrázok 1: Formát hlavičky IPv4[1]



Obrázok 2: Formát hlavičky IPv6[2]

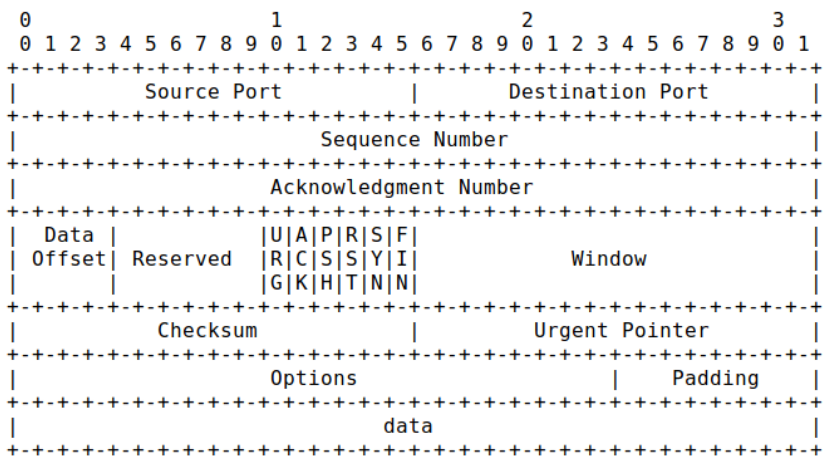
2.2 TCP a SYN skenovanie

Podobne ako IP hlavičku bolo treba vytvoriť a vyplniť hlavičku TCP. SYN skenovanie narozdiel od TCP skenovania neprevádza kompletný 3-way-handsake. Reakcia na na SYN paket môže nadobúdať tri podoby:

1. Odpoveď paketom s nastavenými ACK a SYN flagom - port je otvorený

2. Odpoveď paketom s nastaveným `RST` flagom - port je zatvorený

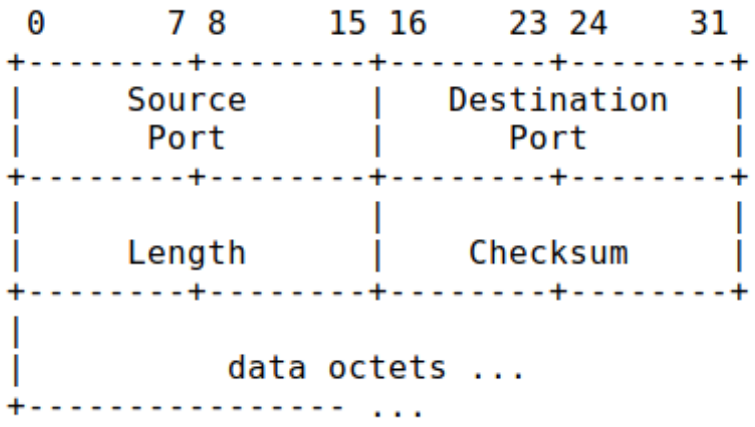
3. Žiadna odpoveď - port považujeme za filtrovaný



Obrázok 3: Formát hlavičky `TCP`[3]

2.3 `UDP` a `UDP` skenovanie

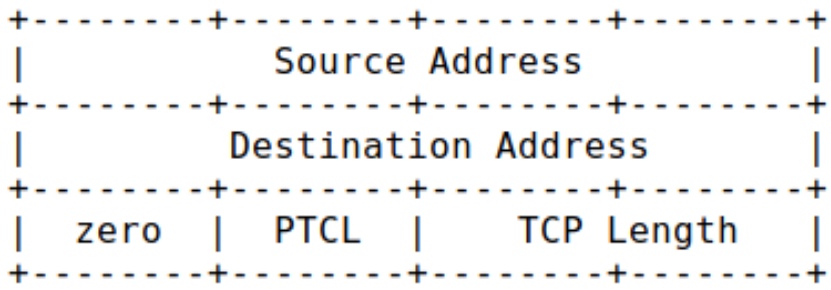
Keďže UDP nenaväzuje spojenie, nie je možné využiť postup ako pri `SYN` skenovaní. Namiesto toho využijeme fakt, že v prípade uzatvoreného portu, systém odpovedá `ICMP` správou `3 port unreachable`. Porty, ktoré touto správou neodpovedajú považujeme za otvorené. Nedostatkom tejto metódy je fakt, že pokiaľ je port filtrovaný, `ICMP` správa sa neodošle a aplikácia teda mylne označí port za otvorený.



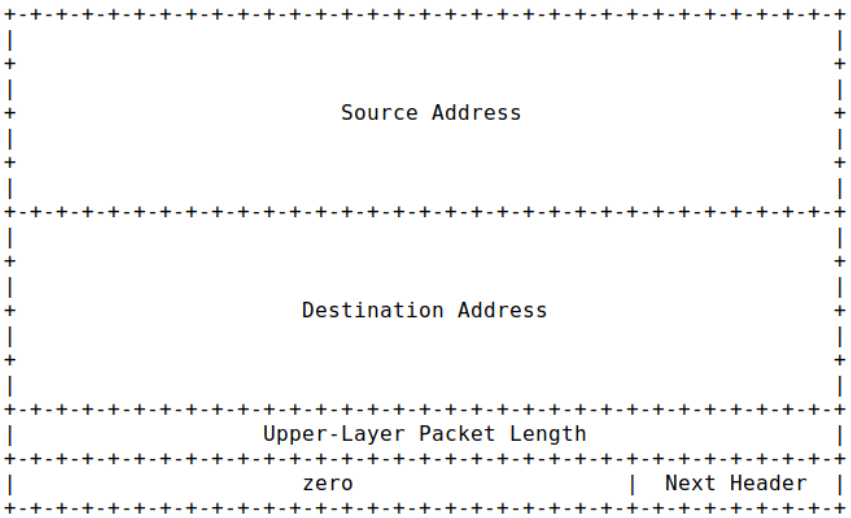
Obrázok 4: Formát hlavičky `UDP`[4]

3 Implementácia

Väčšina implementácie UDP aj SYN skenovania je podobná. Po spracovaní argumentov sa začína vytvárať IP hlavička. Na obrázku 1 a 2 môžeme vidieť, že hlavičky pre IPv4 a IPv6 sa značne líšia. Verzia protokolu je vybraná podľa verzie IP adresy na vstupe. Použité hlavičkové súbory sú definované v súboroch <netinet/ip.h> a <netinet/ip6.h>. V pamäti za IP hlavičku doplníme hlavičku UDP/TCP. Zaujímavou položkou v hlavičkách UDP/TCP je checksum. Presný výpočet tejto položky je popísaný v RFC 1071[5]. Pri implementácií bolo dôležité si dať pozor najme na rozdiel v pseudo hlavičkách medzi IPv4 a IPv6. Na obrázku 5 môžeme vidieť formát pseudo hlavičky pri IPv4 a na obrázku 6 formát pseudo hlavičky pri IPv6.



Obrázok 5: Formát pseudo hlavičky UDP/TCP IPv4[3]



Obrázok 6: Formát pseudo hlavičky UDP/TCP IPv6[2]

So správne vypočítaným checksum zaradíme TCP/UDP hlavičku za IP hlavičku a odošleme paket pomocou BSD sockets. Na zachytávanie odpovedí je využitá knižnica libpcap. Za využitia jej filtrov príjmame a analyzujeme prijaté pakety podľa teórie v v sekcií 2. Timeout pri opakovanom odosielaní paketov je riešený pomocou signálu SIGALRM.

4 Testovanie

Testovanie funkcionality prebiehalo pomocou nástroju `Telnet`, `Wireshark`, `IPv6 Online Port Scanner` (`http://www.ipv6scanner.com/cgi-bin/main.py`). Podstata testovania bola v zachytávaní provozu nástrojom `Wireshark`, sledovaní odoslaných paketov a overovanie ich správneho formátu a pozorovaní obdržaných paketov a následné porovnávanie zistení z `Wiresharku` s výsledkami našej aplikácie. Finálna fáza testovania spočívala vo využití nástroja `IPv6 Online Port Scanner` a porovnávaní jej výsledkov s našou aplikáciou. S dôvodou obmedzených prostriedkov bol testovaný takmer výhdradne localhost.

5 Možnosti spustenia

```
$ ./ipk-scan {-i <interface>} -pu <port-ranges> -pt <port-ranges> [<domain-name> | <IP-address>]
```

- -pt port-ranges skenované `TCP` porty
- -pu port-ranges skenované `UDP` porty
- domain-name — IP-address doménové meno alebo IP adresa skenovaného stroja `TCP` porty
- -i interface identifikátor rozhrania

Príklad použitia:

```
$ ./ipk-scan -pt 21,22,23,25,53,80,110,137,443 localhost -i lo
$ ./ipk-scan -pt 80-100 -pu 53 1.1.1.1
$ ./ipk-scan -pt 80 2001:67c:1220:c1d0:bcf7:27ed:9d5f:c75 -i lo
```

Reference

[1] Rey, M. del. *INTERNET PROTOCOL*. [b.m.]: RFC Editor, September 1981. 1-45 s. RFC, 791. Dostupné na: <https://tools.ietf.org/html/rfc791>.

[2] Deering, S. a Hinden, R. *Internet Protocol, Version 6 (IPv6) Specification*. [b.m.]: RFC Editor, Júl 2017. 1-42 s. RFC, 8200. Dostupné na: <https://tools.ietf.org/html/rfc8200>.

[3] Rey, M. del. *TRANSMISSION CONTROL PROTOCOL*. [b.m.]: RFC Editor, September 1981. 1-85 s. RFC, 793. Dostupné na: <https://tools.ietf.org/html/rfc793>.

[4] Postel, J. *USER DATAGRAM PROTOCOL*. [b.m.]: RFC Editor, August 1980. 1-3 s. RFC, 768. Dostupné na: <https://tools.ietf.org/html/rfc768>.

[5] Braden, R. a Borman, D. *omputing the Internet Checksum*. [b.m.]: RFC Editor, September 1988. 1-24 s. RFC, 1071. Dostupné na: <https://tools.ietf.org/html/rfc1071>.

[6] *Port scanner*.

[7] Fyodor. *The Art of Port Scanning*. Dostupné na: <https://nmap.org/nmap_doc.html>.

[8] *Population density (people per sq. km of land area)*. Dostupné na: <https://data.worldbank.org/indicator/EN.POP.DNST?end=2017&start=2017&type=points&view=map&year_high_desc=true>.

[9] Postel, J. *INTERNET CONTROL MESSAGE PROTOCOL*. [b.m.]: RFC Editor, September 1981. 1-21 s. RFC, 792. Dostupné na: <https://tools.ietf.org/html/rfc792>.

Medzi ďalšie využité zdroje patrili slajdy a example zdrojové súbory k predmetom IPK/ISA na FIT VUT.