



Síťové aplikace a správa sítí

Dokumentácia k projektu
Přenos souboru skrz skrytý kanál

Autor: Samuel Olekšák (xoleks00)
Akademický rok: 2021/22

Obsah

1	Zadanie	2
2	Návod na použitie	3
2.1	Obsah odovzdaného archívu	3
2.2	Použitie	3
2.3	Testovanie	3
3	Teória	4
3.1	ICMP	4
3.2	Zachytávanie paketov	4
3.3	Analýza paketov	5
3.3.1	Ethernetový rámec	5
3.3.2	Hlavička IPv4 paketu	5
3.3.3	Hlavička IPv6 paketu	5
4	Implementácia	6
4.1	Využité knižnice	6
4.2	Protokol	7
4.3	Prenos a ukladanie súboru	8
4.4	Šifrovanie a dešifrovanie	8
4.5	Nastavenie pcap	9
4.6	Spracovanie Ethernet hlavičky	9
4.7	Spracovanie hlavičky IP protokolov	9
4.8	Spracovanie ICMP hlavičky	9
5	Bonusové rozšírenia	9
5.1	Testovací skript	9

1 Zadanie

Vytvořte klient/server aplikaci, která umožní přenést soubor skrz skrytý kanál, kde data jsou přenášena uvnitř ICMP Echo-Request/Response zpráv. Soubor musí být před přenosem zašifrován, aby nebyl přenášen v textové podobě.

Spuštění aplikace: `secret -r <file> -s <ip|hostname> [-l]`

- `-r <file>` : specifikace souboru pro přenos
- `-s <ip|hostname>` : ip adresa/hostname na kterou se má soubor zaslat
- `-l` : pokud je program spuštěn s tímto parametrem, jedná se o server, který naslouchá příchozím ICMP zprávám a ukládá soubor do stejného adresáře, kde byl spuštěn.

Upřesnění zadání: Program zpracuje vstupní argumenty, načte soubor, zašifruje ho a zašle skrz ICMP zprávy na zvolenou IP adresu, kde program, spuštěný v listen (-l) módu, tyto zprávy zachytí, dešifruje a soubor uloží na disk.

- Program může používat pouze ICMP zprávy Echo-request/reply.
- Pro správné chování bude třeba definovat protokol pro přenos dat. (např. je třeba zaslat jméno souboru, ověřit, že soubor byl přenesen celý, apod.) Tento protokol je na vašem uvážení a definujte ho v rámci dokumentace.
- Jako šifru použijte AES, dostupnou např. pomocí knihovny openssl¹. Jako klíč použijte svůj login.
- Program se musí vypořádat se souborem větší, jak max. velikost paketu na standardní síti (1500B), tj. musí být schopen větší soubor rozdělit na více paketů.
- Můžete uvažovat, že v rámci přenosu nedojde ke ztrátám paketů. Pokud implementujete formu spolehlivého přenosu, uveďte to v dokumentaci.
- Při vytváření programu je povoleno použít hlavičkové soubory pro práci se sokety a další obvyklé funkce používané v síťovém prostředí (jako je `netinet/*`, `sys/*`, `arpa/*` apod.), knihovny pro práci s vlákny (`pthread`), pakety (`pcap`), signály, časem, stejně jako standardní knihovnu jazyka C (varianty ISO/ANSI i POSIX), C++ a STL a knihovnu SSL. Další knihovny nejsou povoleny.

¹https://wiki.openssl.org/index.php/EVP_Symmetric_Encryption_and_Decryption

2 Návod na použitie

2.1 Obsah odovzdaného archívu

- `test/` : priečinok s testovacími súbormi,
- `Makefile` : s cieľmi `all` (zostaví skript `secret.cpp`), `clean` (vymaže binárny súbor a archív) a `pack`, (zabalí celý projekt do `.tar` súboru),
- `manual.pdf` : súbor s dokumentáciou,
- `secret.1` : manuálová stránka skriptu,
- `secret.cpp` : zdrojový súbor `secret` programu,
- `test.py` : skript na automatizované testovanie.

2.2 Použitie

Na komunikovanie po zabezpečenom kanále medzi dvomi stranami je potrebné najprv spustiť program `secret` zostavený príkazom `make` na oboch komunikujúcich zariadeniach. Strana, ktorá súbor odosiela je identifikovaná ako klient a spúšťa príkaz `secret` s prepínačom `-s` nasledovaným adresou servera (IPv4 adresa, IPv6 adresa alebo doménové meno) a prepínačom `-r` nasledovaným cestou k súboru na odoslanie. Strana, ktorá súbor prijíma sa nazýva server a musí spustiť príkaz `secret` s prepínačom `-l` pred tým ako sa mu klient pokúsi poslať súbor. Server môže počas jedného spustenia prijímať viacero súborou od jedného alebo viacerých odosielateľov. Súborny sa na serveri ukladajú do rovnakého priečinka ako ten v ktorom bol spustený príkaz `secret -l`, pričom názov súboru sa počas prenosu uchováva. Prípadne, už existujúce súborny s rovnakým menom v tomto priečinku budú prepísané.

Návod na spustenie sa dá vypísať tiež spustením `secret` s prepínačom `-h`, alebo príkazom `man -l secret.1`.

2.3 Testovanie

Spustenie skriptu na automatizované testovanie je popísaný v kapitole 5.1.

3 Teória

3.1 ICMP

Internet Control Message Protocol (ICMP) je používaný hosťiteľmi a smerovačmi na výmenu informácií o sieťovej vrstve. Najtypickejšie použitie ICMP je na oznamovanie chýb. Správy ICMP majú v sebe polia na typ a kód a obsahujú hlavičku a prvých 8 bajtov IP datagramu, ktorý vyvolal ICMP správu. [5]

ICMP Type	Code	Description
0	0	echo reply (to ping)
3	0	destination network unreachable
3	1	destination host unreachable
3	2	destination protocol unreachable
3	3	destination port unreachable
3	6	destination network unknown
3	7	destination host unknown
4	0	source quench (congestion control)
8	0	echo request
9	0	router advertisement
10	0	router discovery
11	0	TTL expired
12	0	IP header bad

Obr. 1: Typy a kódy ICMP paketov[5]

3.2 Zachytávanie paketov

Zachytávač paketov (packet sniffer) je pasívny prijímač, ktorý zaznamenáva kópiu každého paketu, ktorý prechádza okolo. Zachytené pakety sa následne dajú analyzovať, aj na citlivé informácie. Software na zachytávanie paketov je voľne dostupný na rôznych webových stránkach a v komerčne dostupných produktoch. Keďže je zachytávanie paketov pasívne – čiže, zachytávače nevkladajú pakety do kanálu – sú zachytávače náročné na detekciu. Takže ak posielame pakety do bezdrôtového kanálu, musíme akceptovať možnosť, že pakety môžu byť zachytávané a ukladané niekým iným. Na zabránenie zneužitia sa používajú riešenia zahrňujúce kryptografiu.[5]

Na zachytávanie paketov v tomto projekte je využitá prenosná C/C++ knižnica na zachytávanie sieťovej premávky `libpcap`. [4]

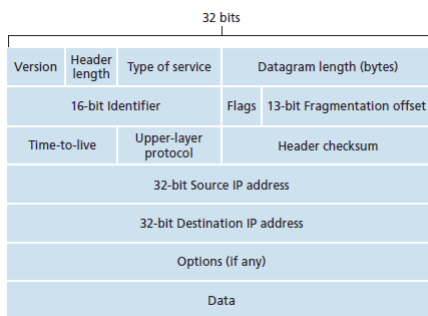
3.3 Analýza paketov

3.3.1 Ethernetový rámec

V ethernetovom rámci má hlavička fixnú dĺžku 14 bytov – na prvých 12 bytoch je uložená zdrojová a cieľová MAC adresa a v nasledujúcich 2 bytoch je udaný Ethertype – identifikácia protokolu vyššej vrstvy.[1]

3.3.2 Hlavička IPv4 paketu

IPv4 datagram môže obsahovať variabilné množstvo nastavení, preto položka *Header length* obsahujúca 4 bity je potrebná na určenie, kde sa končí hlavička a začínajú dáta IPv4 datagramu. Väčšina IP datagramov neobsahuje nastavenia, preto je typická dĺžka IPv4 hlavičky 20 bytov.[6] Minimálna hodnota tohto poľa je 5, čo indikuje dĺžku $5 \times 32 \text{ bitov} = 160 \text{ bitov} = 20 \text{ bytov}$. [3] Všetky položky hlavičky sú zobrazené na obr. 2.

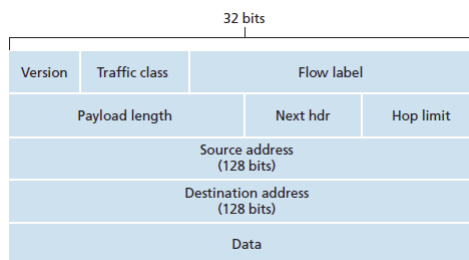


Obr. 2: Štruktúra IPv4 paketu[5]

3.3.3 Hlavička IPv6 paketu

Formát IPv6 datagramu je ukázaný na obr. 3. Naproti IPv4 má IPv6 hlavička fixnú dĺžku 40 bytov. Položka *Next hdr* indikuje protokol vyššej vrstvy tohto datagramu. Zdrojová a cieľová adresa má 32 bitov jej formát je bližšie popísaný v RFC 4291².

²<https://tools.ietf.org/html/rfc4291>



Obr. 3: Štruktúra IPv6 paketu^[5]

4 Implementácia

4.1 Využívané knižnice

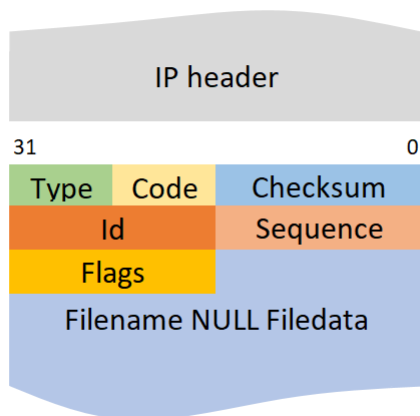
- `openssl/aes.h` – AES šifrovanie,
- `pcap/pcap.h` – vysokoúrovňové rozhranie na zachytávanie paketov³,
- `pcap/sll.h` – spracovanie SLL hlavičky⁴,
- `getopt.h` – spracovanie argumentov príkazového riadka,
- `arpa/inet.h` – konvertovanie IP adres do čitateľnej podoby,
- `netinet/ether.h` – spracovanie ethernetovej hlavičky,
- `netinet/ip.h` – spracovanie IPv4 hlavičky,
- `netinet/ip6.h` – spracovanie IPv6 hlavičky,
- `netinet/ip_icmp.h` – spracovanie ICMP hlavičky,
- `netinet/icmp6.h` – spracovanie ICMPv6 hlavičky
- `netdb.h` – preklad doménového mena na IP adresu.

³<https://www.tcpdump.org/manpages/pcap.3pcap.html>

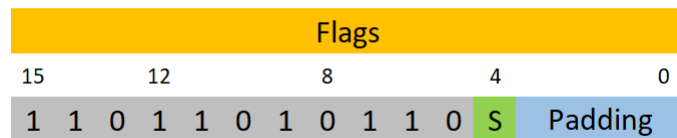
⁴<https://wiki.wireshark.org/SLL>

4.2 Protokol

ICMP hlavička obsahuje typ a kód korešpondujúci správe echo request. Id a číslo sekvencie z ICMP echo hlavičky nie sú využité keďže tieto polia sa môžu meniť v sieti v priebehu preposiľania paketu. Nasleduje 16-bitové pole príznakov (obr. 4) - 11 fixných bitov, ktoré slúžia ako identifikácia tohto protokolu a odfiltrovanie iných ICMP paketov na serveri (server kontroluje toto pole a ak sa identifikácia nezhoduje, paket zahodí), 1 start bit, ktorý značí, či je tento paket prvý pre daný súbor a 4 bity, ktoré označujú množstvo výplne (padding) na konci súborových dát. Nasleduje pole obsahujúce názov súboru, ktorý sa prenáša. Názov súboru je obsiahnutý v každom pakete s dátami daného súboru (týmto sa pakety rôznych súborov navzájom identifikujú). Dĺžka názvu súboru je veľmi variabilná a na drvivej väčšine bežne používaných súborových systémoch obmedzená na 255 znakov a nesmie obsahovať NULL znaky. V protokole je názov súboru oddelený od zašifrovaných súborových dát znakom NULL.



Obr. 4: Schéma paketu



Obr. 5: Schéma `flags` poľa hlavičky - fixných 11 bitov, start bit a 4 bity udávajúce výplň na konci súborových dát

4.3 Prenos a ukladanie súboru

Keďže zadanie hovorí, že prenosový kanál je spoľahlivý - nedochádza k stratám a zmene poradia paketov, môžeme do súboru dáta paketov postupne zapisovať v móde `append`. Prvý paket súboru má v príznakoch nastavený príznak **S** (**start**), podľa ktorého server vie, že dáta tohto prvého paketu majú byť zapísane v móde `rewrite`, ktorý prepisuje súbor, ak nejaký s rovnakým menom už existuje, nasledujúce pakety do súboru iba pripisujú (`append`). V prípade, že by v kanále mohlo dochádzať k zmenám poradia paketov, bolo by možné pridať si nové pole udávajúce poradie paketu v rámci súboru (ideálne minimálne 32 bitov, aby pri veľkých súboroch nedošlo k pretečeniu). Na základe poradia a známeho (konštantného) množstva dát v každom pakete (`MAX_ICMP_DATA_SIZE` - dĺžka názvu súboru - 1 (NULL bajt oddelujúci tieto dve časti)) by bolo možné vypočítať, kde do výstupného súboru tieto dáta patria. Čiže v prípade, že by prišiel 2. paket súboru pred prvým paketom súboru vznikol by riedky súbor až do chvíle kým príde 1. paket, ktorý toto prázdne miesto vyplní. Program nezachováva adresárovú štruktúru odoslaných dát, čiže nehľadá na cestu k odosielanému súboru, iba na jeho názov.

4.4 Šifrovanie a dešifrovanie

Na šifrovanie a dešifrovanie je použitá 128-bitová bloková šifra s pomocou knižnice `openssl/aes.h`. Keďže veľkosť bloku je 16 bajtov, bolo potrebné veľkosť posielaných dát v bajtoch stále zaokrúhliť na najbližší vyšší násobok 16, pričom prázdne miesta boli vyplnené NULL. Pre každý paket sa posiela v hlavičke počet vyplňovacích bajtov, aby sa po dešifrovaní tieto NULL znaky nezapísali do preneseného súboru. Ako šifrovací kľúč je použitý duplicitný login `xoleks00xoleks00`, aby bola dĺžka kľúča 16 bajtov.

4.5 Nastavenie pcap

Knižnica pcap umožňuje zachytávať a vypisovať pakety v reálnom čase po zapnutí *immediate* módu pomocou funkcie `pcap_set_immediate_mode`. Buffer na prichádzajúce pakety je nastavený na veľkosť 256 MiB funkciou `pcap_set_buffer_size`, aby bol schopný prijať dostatočne veľké súbory. Na odfiltrovanie paketov iných ako ICMP echo správy je použitý filter `"icmp[icmptype]=icmp-echo or icmp6[icmp6type]=icmp6-echo"`.

4.6 Spracovanie Ethernet hlavičky

Na jednoduchšie spracovanie Ethernet hlavičky poskytuje knižnica `net/ethernet.h` štruktúru `ether_header`. Po pretypovaní ukazovateľa na obsah paketu na ukazovateľ na túto štruktúru sa dá extrahovať typ paketu vyššej vrstvy, ktorý je dôležité poznať pred nasledujúcim krokom. Rovnaká knižnica taktiež poskytuje konštanty reprezentujúce hodnoty jednotlivých typov ethernetového paketu[2]:

- `ETHERTYPE_IP` (0x0800) – IPv4 paket,
- `ETHERTYPE_IPV6` (0x86DD) – IPv6 paket.

4.7 Spracovanie hlavičky IP protokolov

Podľa položky `ethertype` z predošlej hlavičky vieme určiť protokol transportnej vrstvy. Vďaka pcap filtru však máme zaručené, že zachytávame iba ICMP pakety. Knižnice `netinet/ip` a `netinet/ip6` poskytujú štruktúry na jednoduché vybratie IP adresy z IP hlavičky `iphdr` a `ip6_hdr`.

4.8 Spracovanie ICMP hlavičky

Knižnice `netinet/icmp6.h` a `netinet/ip_icmp.h` poskytujú štruktúry na jednoduché rozobratie ICMP hlavičky. Polia `identification` a `sequence` nie sú použité. Spracovanie nasledujúcich dát je popísané v kapitole 3.2.

5 Bonusové rozšírenia










5.1 Testovací skript

V archíve je priložený testovací skript `test.py` spolu s priečinkom `test/` obsahujúcim súbory na testovanie. Skript pre každý z testov spustí server a následne klienta spolu s testovacím súborom a testovacou IP adresou (odkazujúcou sa loopback alebo lokálnu adresu). Po uplynutí niekoľkých sekúnd sa server zastaví a prijatý súbor

sa porovná so súborom, ktorý sa odoslal a ak sa zhodujú, test je úspešný. Zdrojový kód testovacieho skriptu je vytvorený tak, aby bol veľmi ľahko rozšíriteľný o ďalšie testy.

Jednotlivé testy sa medzi sebou líšia v charaktere odoslaného súboru (zmestí sa do jedného paketu/viacerych paketov, typ súboru (plain text, binárne dáta), obsahuje Unicode znaky atp.) a v type adresy (IPv4, IPv6, doménové meno).

Príklad spustenia: `sudo python3 test.py`

Name	Size
 test1.txt	6 bytes
 test2.txt	23 bytes
 test3.txt	26 bytes
 test4.txt	41 bytes
 test5.png	544 bytes
 test6.png	3,8 kB
 test7.txt	5,0 kB
 test8.jpg	83,9 kB
 test9.jpg	679,8 kB

Obr. 6: Zoznam priložených testov s postupne rastúcou veľkosťou

```
student@student-vm:~/ISA$ sudo python3 test.py
[sudo] password for student:
TEST 1 - It should send tiny plain text file to provided IPv4 loopback: PASSED
TEST 2 - It should send tiny plain text file with special characters to provided IPv4 local address: PASSED
TEST 3 - It should send tiny plain text file to localhost hostname translated to IPv4 address: PASSED
TEST 4 - It should send tiny plain text file to IPv6 local address: PASSED
TEST 5 - It should send tiny image to IPv6 loopback address: PASSED
TEST 6 - It should send small image in multiple packets: PASSED
TEST 7 - It should send plain text file in multiple packets: PASSED
TEST 8 - It should send large image in multiple packets: PASSED
TEST 9 - It should send huge image in multiple packets using IPv6 without running out of buffer: PASSED
=====
PASSED 9/9
```

Obr. 7: Výstup testovacieho skriptu

Zdroje

- [1] *Ethernet frame*, online. URL: https://www.wikiwand.com/en/Ethernet_frame.
- [2] *IEEE 802 Numbers*, online. URL: <https://www.iana.org/assignments/ieee-802-numbers/ieee-802-numbers.xhtml>.
- [3] *IPv4*, online. URL: <https://en.wikipedia.org/wiki/IPv4>.
- [4] *Man page of PCAP*, online. URL: <https://www.tcpdump.org/manpages/pcap.3pcap.html>.
- [5] J. Kurose, J. F. a K. W. Ross, *Computer networking: a top-down approach*, 7. vyd. Pearson, 2017, ISBN: 0133594149.
- [6] J. Garud, *IPv4 Datagram Format*, online, máj 2016. URL: <https://electronicspost.com/ipv4-datagram-format/>.