VYSOKÉ UČENÍ TECHNICKÉ V BRNĚ
BRNO UNIVERSITY OF TECHNOLOGY

FAKULTA INFORMAČNÍCH TECHNOLOGIÍ
FACULTY OF INFORMATION TECHNOLOGY

TFTP KLIENT + SERVER

SEMESTRÁLNÍ PROJEKT
TERM PROJECT

AUTOR PRÁCE
AUTHOR

JOSEF KUCHAŘ (XKUCHA28)

BRNO 2023

Obsah

Kapitola 1

Uvedení do problematiky

1.1 Zadání

Cílem projektu bylo implementovat TFTP klient a server dle korespondujícího RFC [6]. Dále bylo za úkol implementovat rozšíření

- TFTP Option Extension [4]
- TFTP Blocksize Option [3]
- TFTP Timeout Interval and Transfer Size Options [5]

viz zadání [2].

1.2 Základní popis protokolu

TFTP, neboli Trivial File Transfer Protocol, je jednoduchý protokol pro přenos souborů mezi počítači v počítačové síti. Používá UDP (User Datagram Protocol) [1] pro přenos dat a často slouží k přenosu firmware, konfiguračních souborů nebo bootovacích souborů mezi zařízeními.

1.3 Limitace protokolu

V základní formě neposkytuje žádnou formu šifrování nebo autentizace, což z něj činí méně vhodnou volbu pro přenos citlivých dat přes veřejné sítě, jako je internet, proto je využíván hlavně v sítích LAN.

Při komunikaci pomocí TFTP se náhodně vybírají porty pro přenos dat. Některé firewally mohou být konfigurovány tak, aby povolily pouze komunikaci na určitých portech, což může být konfliktní s náhodným výběrem portů, který se děje v tomto protokolu.

Kapitola 2

Návrh aplikace

Návrh aplikace pevně vyplývá ze zadání a příslušných RFC. V případě ztraceného/opožděného paketu je komunikace opakována.

2.1 Návrh klienta

Klient může od libovolného TFTP serveru stáhnout soubor, nebo mu nějaký soubor zaslat. V případě chyby komunikace ukončuje spojení, maže případný nedokončený soubor a končí chybovou návratovou hodnotou.

2.2 Návrh serveru

Server je schopný obsluhovat více požadavků současně. V případě vadného požadavku, nebo pokud nastane chyba během komunikace, ukončí komunikaci a dál pracuje normálně. Cílové soubory z důvodu bezpečnosti nelze mazat ani přepisovat.

Kapitola 3

Popis implementace

3.1 Obecné informace o implementaci

Klient i server je napsaný v jazyce C++ 17. Je implementovaná veškerá funkcionalita požadovaná v zadání [2].

Signál CTRL+C je korektně ošetřený na obou stranách. Opačné straně je zaslán `ERROR` packet o ukončení programu a případné nedokončené přenášené soubory jsou smazány.

3.2 Implementace serveru

Ke konkurentní obsluze požadavků od klientů využívá `std::thread`.

3.3 Odevzdané soubory

- `utils.cc` a `utils.h` - obsahují abstrakce nad funkcemi pro příjímání a odesílání UDP packetů, poskytují funkce pro práci s `netascii` kódováním
- `enums.h` a `settings.h` - obsahuje definici enumů pro typy packetů, typy chyb, velikost bloku a podobně
- `packet.cc` a `packet.h` - kolekce struktur a funkcí pro parsování a zobrazování TFTP paketů
- `packet-builder.cc` a `packet-builder.h` - implementuje třídu, která abstrahuje manuální tvoření packetů.
- `client-args.cc` a `client-args.h` - implementuje třídu pro parsování argumentů z příkazové řádky pro TFTP klienta
- `tftp-client.cc` a `tftp-client.h` - samotná implementace TFTP klienta
- `server-args.cc` a `server-args.h` - implementuje třídu pro parsování argumentů z příkazové řádky pro TFTP server
- `tftp-server.cc` a `tftp-server.h` - samotná implementace TFTP serveru

Kapitola 4

Základní informace o programu

Klient i server vypisují na standardní chybový výstup informace o příchozích paketech dle zadání [2]:

- `RRQ {SRC_IP}:{SRC_PORT} "{FILEPATH}" {MODE} {$OPTS}`
- `WRQ {SRC_IP}:{SRC_PORT} "{FILEPATH}" {MODE} {$OPTS}`
- `ACK {SRC_IP}:{SRC_PORT} {BLOCK_ID}`
- `OACK {SRC_IP}:{SRC_PORT} {$OPTS}`
- `DATA {SRC_IP}:{SRC_PORT}:{DST_PORT} {BLOCK_ID}`
- `ERROR {SRC_IP}:{SRC_PORT}:{DST_PORT} {CODE} "{MESSAGE}"`

Jednotlivé extension options `{$OPTS}` pak ve formátu dle pořadí v datovém přenosu:

- `{OPT1_NAME}={OPT1_VALUE} ... {OPTn_NAME}={OPTn_VALUE}`

Pro přehlednost komunikace vypisuje program na standardní výstup i odchozí pakety ve stejném formátu jako příchozí pakety s tím rozdílem, že je uvozuje řetězec `>>`, aby bylo jasné, které pakety jsou příchozí a které odchozí.

V případě neúspěšného přenosu je soubor v cílové destinaci smazán. Soubory na straně serveru nelze přepisovat, na straně klienta lze, protože se to netýká TFTP specifikace.

4.1 Základní informace o klientovi

Klient k příjímání i odesílání souborů používá formát `octet` viz [6].

Klient nepoužívá žádné rozšíření a komunikuje pouze podle základní specifikace.

Z důvodu použití standardního vstupu pro posílání souborů na server klient nemůže posílat celkovou velikost souboru, ani před odesláním ověřit, zda je soubor dostatečně malý.

4.2 Základní informace o serveru

Server podporuje formáty přenosu `octet` a `netascii` viz [6]. Formát `mail` nebyl implementován z důvodu zastaralosti.

Kapitola 5

Návod na použití

Návod na použití programu vychází přímo ze zadání [2].

Pro testovací účely lze spustit server pomocí `make run_server` a následně pomocí `make run_client_send` odeslat testovací soubor z klienta na server, případně lze spustit `make run_client_recv` a tím stáhnout testovací soubor ze serveru pomocí klienta. Po obou těchto operacích je pomocí programu `diff` porovnán obsah souborů pro ověření koretktního přenosu.

5.1 Návod na použití klienta

```
tftp-client -h hostname [-p port] [-f filepath] -t dest_filepath
```

- `-h` IP adresa/doménový název vzdáleného serveru
- `-p` port vzdáleného serveru, pokud není specifikován je použit port 69
- `-f` cesta ke stahovanému souboru na serveru (download), pokud není specifikován používá se obsah stdin (upload)
- `-t` cesta, pod kterou bude soubor na vzdáleném serveru/lokálně uložen

5.2 Návod na použití serveru

```
tftp-server [-p port] root_dirpath
```

- `-p` místní port, na kterém bude server očekávat příchozí spojení
- cesta k adresáři, pod kterým se budou ukládat příchozí soubory

Literatura

[1] *User Datagram Protocol* [RFC 768]. RFC Editor, srpen 1980. DOI: 10.17487/RFC0768. Dostupné z: https://www.rfc-editor.org/info/rfc768.

[2] Dolejška, D. *TFTP Klient + Server* [[online]]. [vid. 2023-11-09]. Dostupné z: https://www.vut.cz/studis/student.phtml?script_name=zadani_detail&apid=268266&zid=54264.

[3] Malkin, G. S. a Harkin, A. *TFTP Blocksize Option* [RFC 2348]. RFC Editor, květen 1998. DOI: 10.17487/RFC2348. Dostupné z: https://www.rfc-editor.org/info/rfc2348.

[4] Malkin, G. S. a Harkin, A. *TFTP Option Extension* [RFC 2347]. RFC Editor, květen 1998. DOI: 10.17487/RFC2347. Dostupné z: https://www.rfc-editor.org/info/rfc2347.

[5] Malkin, G. S. a Harkin, A. *TFTP Timeout Interval and Transfer Size Options* [RFC 2349]. RFC Editor, květen 1998. DOI: 10.17487/RFC2349. Dostupné z: https://www.rfc-editor.org/info/rfc2349.

[6] Sollins, D. K. R. *The TFTP Protocol (Revision 2)* [RFC 1350]. RFC Editor, červenec 1992. DOI: 10.17487/RFC1350. Dostupné z: https://www.rfc-editor.org/info/rfc1350.