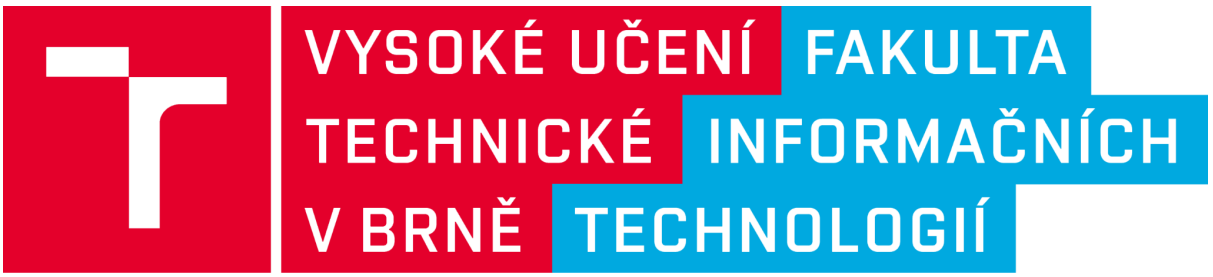

Dokumentace k projektu z předmětu ISA

Varianta: TFTP Klient + Server

autor: Ondřej Bahounek, xbahou00

Obsah

1. Úvod

Cílem projektu bylo vytvořit server a klientskou aplikaci, které jsou schopné sdílet soubory, přes TFTP (Trivial file transfer protocol). Klient i server vypisují přijatou komunikaci na výstup.

2. TFTP protokol

TFTP (trivial file transfer protocol) je jednoduchý protokol umožňující klientovy nahrát soubory na server nebo soubory na server nahrát. TFTP je implementována nad protokolem UDP.
TFTP komunikace začíná tím že klient pošle žádost o nahrání nebo stažení souboru na server.
V případě žádosti o stažení souboru, server začne posílat soubor po blokách o velikosti 512 bajtů. Klient na každý přijatý blok, odešle potvrzení o přijetí bloku s příslušným číslem. Až po přijetí potvrzení server pošle další blok.
V případě žádosti o nahrání, server odešle potvrzení o přijetí bloku číslo nula.
Přenes je ukončen pokud jedna ze stran přijme blok s velikostí menší než 512 bajtů.

3. Návrh programu a implementace

3.1 Zpracování parametrů

Klient i server zpracovávají vstupní argumenty voláním funkce **parse_arguments()**, která s využitím funkce getopt() z knihovny <getopt.h> nahraje informace do datové struktury **command_line_args_t**, kterou funkce následně vrátí.

3.2 Klient začátek komunikace

Klient si vytváří schránku (socket) a na IP adresu a port daný parametry programu odesílá žádost o sdílení souboru. Pokud nebylo zadané číslo portu, klient odesílá paket na výchozí port pro TFTP komunikaci 69.

3.3 Server čeká na žádost

Server po spuštění čeká na portu 69, pokud nebylo parametrem zadáno jinak na příchozí žádost. Server čeká na příchozí zprávu pomocí funkce **recvfrom()** z knihovny <sys/socket.h>, po přijetí zprávy server vytvoří nový proces pomocí funkce **fork()** z <stdlib.h>. Otcovský proces je zastaven zamčený semaforem, který odblokuje až dětský proces poté co zkopíruje údaje o odesílateli. Otcovský proces jde následně opět čekat na další příchozí paket. Dětský proces následně volá funkci **handle_first_packet()**, která pokračuji v komunikaci.

3.4 Server od odpověď na žádost

Server načte a zpracuje příchozí žádost do datové struktury **initial_data_t**. Struktura obsahuje také informace o tom jestli klient vyžaduje nějaká serverem podporovaná rozšíření nad rámec základní specifikace TFTP. Server podporuje rozšíření blksize, tsize and timeout dle jejich specifikace (viz. použitá literatura). Server si pro zbytek komunikace otevřené nový schránku a zprávy odesílá z nového (náhodného) portu, který slouží jako jeho identifikátor, pro zbytek komunikace . Server na žádost o stažení souboru bez požadovaných rozšíření odpovídá odesláním prvního DATA TFTP paketu. Pokud klient požadoval rozšíření server odešle buď TFTP packet 0ACK s seznamem rozšíření, které server podporuje nebo DATA paket, čímž odmítne všechny rozšíření. Klient má možnost zamítnout vyjednaná rozšíření ERROR zprávou nebo je přijmout DATA blockem s číslem bloku nastaveným na 0. Na žádost o nahrání souboru na server, server reagujem buď odesláním ACK paketu s číslem bloku nastaveným na 0 nebo 0ACK specifikující přijaté rozšíření.

3.5 Průběh a ukončení komunikace

Klient čeká na odpověď pomocí funkce recvfrom(), protože server odešle odpověď z jiného portu než na který odeslal klient požadavek. Klient používá nový port serveru pro zbytek komunikace. Sever a klient si posílají soubor po datových blocích o velikosti 512B, pokud se nedohodli jinak. Po odeslání každého bloku dat DATA N odesílatel čekat až mu od příjemce přijde ACK paket, potvrzující přijetí bloku N. Pokud nepřišel očekávaný block dat nebo očekávané potvrzení o přijetí do 3 sekund (pokud není specifikováno jinak) server nebo klient odešle znovu poslední paket a vrátí se znovu k čekání. Poslední paket se odešle až třikrát před odesláním ERROR zprávy a ukončené komunikace a běhu programu. Komunikace je za normálních podmínek ukončená přijetím bloku dat o velikosti menší než 512 a odesláním a přijetím posledního potvrzení.

3.6 Uložení souboru

Příchozí soubor si příjemce postupně nahrává do struktury **data_and_size_t**. Funkce **create_file()** po ukončení komunikace vytvoří nový soubor pojmenovaný dle parametrů programu nebo podle názvu uvedeného v žádosti o přenos. A nahraje do něj všechny přijatá data.

4. Použité knihovny

4.1 Standardní C knihovny

- <stdio.h>, <stdlib.h>, <string.h>, <stdbool.h>, <ctype.h> - základní knihovny pro práci se s řetězci a funkcemi jako malloc
- <getopt.h> - na zpracování vstupních argumentů
- <signal.h> - na zachycení CTRL + C
- <sys/time.h>, <sys/select.h> - umožní čekat na příchozí paket určitý čas
- <unistd.h> - pro fork()
- <semaphore.h>, <fcntl.h>, <sys/stat.h> - semafory na řízení běhu programu
- <errno.h> - zjištění chybového kódu

4.2 Knihovny pro práci se sítí

- <sys/socket.h> - struktura socket

- <arpa/inet.h> - převáděné s network byte order
- <netinet/in.h> - struktury pro ip adresu, port, atd…
- <netdb.h> - gethostbyname()

4.3 Uživatelské knihovny

- “tftp-utils.h”- knihovna obsahující funkce používné v tftp-client.c i tftp-server.c

5. Spuštění programu

Klinet se spouší pomocí příkazu:
./tftp-client -h hostname [-p port] [-f filepath] -t dest_filepath

- hostname je jméno nebo IPv4 adresa serveru
- port je číslo portu kde server bude očekávat komunikaci
- filepath je cesta k souboru, který chci ze serveru stáhnout
    - pokud není zadáno, klient bude nahrávat soubor na server, soubor který bude očekávat na stdin
- dest_path je jméno pod kteým se má soubor uložit

Spuštění serveru:
sudo ./tftp-server [-p port] root_dirpath

- port je číslo portu na kterém bude server očekávat komunikaci
- root_dirpath je adresář do kterého a ze kterého server nahrává soubory

6. Zdroje

[1] Sollins, K., "The TFTP Protocol (Revision 2)", STD 33, RFC 1350 (Online at https://datatracker.ietf.org/doc/html/rfc1350), MIT, July 1992.

[2] Malkin, G., and A. Harkin, "TFTP Option Extension", RFC 2347 (Online at https://datatracker.ietf.org/doc/html/rfc2347), May 1998.

[3] Malkin, G., and A. Harkin, "TFTP Blocksize Option", RFC 2348 (Online at https://datatracker.ietf.org/doc/html/rfc2348), May 1998.

[4] Malkin, G., and A. Harkin, "TFTP Timeout Interval and Transfer Size Options", RFC 2349 (Online at https://datatracker.ietf.org/doc/html/rfc2349), May 1998.