VYSOKÉ UČENÍ TECHNICKÉ V BRNĚ
FAKULTA INFORMAČNÍCH TECHNOLOGIÍ

Síťové aplikace a správa sítí

TFTP Klient + Server

20. listopadu 2023

Samuel Čus (xcussa00)

Obsah

1 Zadání

Cílem projektu byla implementace klientské a serverové aplikace pro přenos souborů prostřednictvím TFTP (Trivial File Transfer Protocol) a to přesně podle korespondující RFC specifikace. Výsledná řešení musí dále být v souladu s následujícími rozšířeními základní specifikace protokolu TFTP:

- TFTP Option Extension
- TFTP Blocksize Option
- TFTP Timeout Interval and Transfer Size Options

2 Trivial File Transfer Protocol

TFTP (Trivial File Transfer Protocol) je protokol pro přenos souborů, který slouží k jednoduchému přenášení dat mezi počítači v počítačových sítích. Byl navržen tak, aby byl co nejjednodušší a měl minimální nároky na implementaci. TFTP se často využívá v situacích, kde je potřeba rychlý a nenáročný přenos souborů, zejména v embedded zařízeních, vestavěných systémech nebo v prostředích s omezenými zdroji. Zaměřuje na základní funkce přenosu souborů a minimalizuje složitost protokolu. Jeho návrh je jednoduchý, což usnadňuje implementaci. Podporuje pouze základní operace čtení (read) a zápis (write) souborů. Nemá rozsáhlé funkce, které najdeme v komplexnějších protokolech pro přenos souborů. Pro přenos dat využívá TFTP protokol UDP, což je protokol bez spojení. Tato volba zjednodušuje implementaci, ale zároveň může znamenat menší spolehlivost než u protokolů s připojením, jako je TCP. TFTP obvykle neposkytuje bezpečnostní prvky, jako jsou šifrování nebo autentizace. To znamená, že data přenášená pomocí TFTP nejsou automaticky chráněna před neautorizovaným přístupem nebo sledováním.

3 Spuštění

3.1 Klient

./tftp-client -h hostname [-p port] [-f filepath] -t destfilepath

- -h IP adresa/doménový název vzdáleného serveru
- -p port vzdáleného serveru - pokud není specifikován předpokládá se výchozí dle specifikace
- -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

3.2 Server

./tftp-server [-p port] rootdirpath

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

4 Návrh

Největší otázkou návrhu bylo jak strukturovat přijaté a odeslané zprávy. Z toho také vyplynula otázka, jestli by nebylo jednodušší implementovat aplikace v C++ a to kvůli OOP. Nakonec jsem se ale rozhodl pro implementaci v jazyce C. K reprezentaci různých typů zpráv v protokolu TFTP se používá union tftp_message. Každý typ zprávy má svůj vlastní formát. Union umožňuje sdílet paměťový prostor mezi různými strukturami, takže může být interpretován různými způsoby na základě aktuálního obsahu. Projekt je rozdělen do 3 částí a to klient, server a messages. Messages implementuje rozhraní sloužící klientovi a serveru k posílání nebo přijímání různých typů zpráv.

5 Implementace

5.1 Klient

Program začíná zpracováním argumentů z příkazové řádky jako hostname serveru, cesta k cílovému souboru, port a další parametry. Pokud nebyl specifikován port, program použije výchozí hodnotu 69, což je standardní port pro TFTP. Po načtení a kontrole argumentů následuje vytvoření UDP socketu pro komunikaci se serverem. Samotný přenos souborů má dvě fáze: UPLOAD (přenos z klienta na server) a DOWNLOAD (přenos ze serveru na klienta). Pro UPLOAD je vytvořena funkce client_send(), která postupně čte data ze vstupu a odesílá je serveru. Tato funkce rovněž kontroluje potvrzení od serveru a v případě neúspěchu opakuje přenos. Pro DOWNLOAD slouží funkce client_receive(), která zajišťuje přenos souboru ze serveru na klienta. Program očekává potvrzení od serveru a přenáší data v případě korektní odpovědi. Obě fáze jsou řízeny funkcemi handle_rrq() a handle_wrq(), které zasílají READ REQUEST nebo WRITE REQUEST zprávy na server. Celková logika programu je v hlavní funkci main, která zajišťuje načítání argumentů, inicializaci síťového spojení s serverem, a volání příslušné fáze přenosu (UPLOAD nebo DOWNLOAD) přes již zmíněné funkce. Po dokončení přenosu dochází k uzavření síťového spojení s TFTP serverem

5.2 Server

Nejprve jsou kontrolovány argumenty příkazové řádky pomocí funkce check_args(), která ověřuje, zda byly zadány správné argumenty. Pokud nebyl specifikován port, program použije výchozí hodnotu 69, což je standardní port pro TFTP. Dále se určí cesta k adresáři, kam budou ukládány přijaté soubory. Následně je vytvořen a nastaven UDP socket pomocí funkcí create_socket() a server_bind(). Socket je vázán na specifikovaný port a čeká na příchozí požadavky. Hlavní smyčka programu je obsažena ve funkci server(). Program v této smyčce neustále čeká na příchozí zprávy od klientů. Jakmile dorazí požadavek na čtení (RRQ) nebo zápis (WRQ), vytvoří se nový proces pomocí fork, který obslouží daný požadavek. Funkce handleclientrqst() zpracovává klientův požadavek. Nejprve se analyzují a zpracovávají případné TFTP options, pokud jsou přítomny. Poté se zkontroluje, zda cesta k souboru není mimo povolený adresář, a podle toho se spustí buď funkce pro stahování (server_download()) nebo nahrávání souboru (server_upload()). Obě tyto funkce provádějí samotný proces přenosu dat mezi klientem a serverem. Funkce pro stahování odesílá požadovaný soubor, zatímco funkce pro nahrávání přijímá data od klienta a ukládá je do souboru. Program je schopen obsluhovat více klientů současně, díky použití procesů vytvořených funkcí fork. Smyčka serveru běží neustále, což umožňuje obsluhu více klientů v průběhu času.

5.3 Messages

Soubor obsahuje sadu funkcí společně používaných serverem a klientem pro komunikaci pomocí protokolu TFTP (Trivial File Transfer Protocol). Jsou zde implementovány funkce reprezentující jednotlivé zprávy používané jak klientem během TFTP přenosu(ACK, OACK, DATA, ERROR). RRQ i WRQ zde implementovány nejsou jelikož jsou využívány pouze klientem. Funkce receive_message() slouží k příjímání a zpracování zpráv a jejich uložení do předem vytvořené union tftp_message, již dříve zmíněné v návrhové části, která je definována v hlavičkovém souboru messages.h. Tam také můžeme najít definice enums pro opcodes, errory a také pro typ přenosu.

Reference

[1] https://datatracker.ietf.org/doc/html/rfc1350

[2] https://datatracker.ietf.org/doc/html/rfc2347

[3] https://datatracker.ietf.org/doc/html/rfc2348

[4] https://datatracker.ietf.org/doc/html/rfc2349

[5] http://www.tcpipguide.com/free/t_TFTPGeneralOperationConnectionEstablishmentandClie.htm