ISA – manuál

TFTP Klient + Server

Michal Ondrejka, xondre15

20.11.2023

Obsah

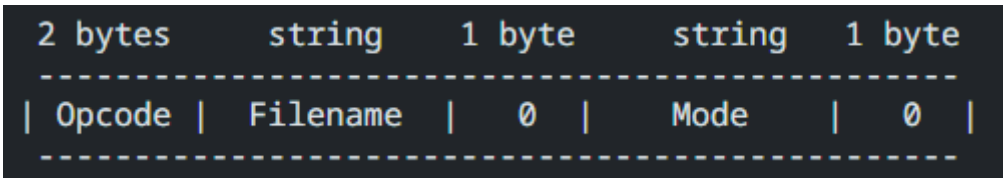
1. Uvedenie do problematiky

Problematika

V dnešnej dobe, kedy sa informácie a dáta pohybujú rýchlosťou svetla, je dôležité mať efektívne a spoľahlivé spôsoby prenosu súborov medzi zariadeniami v počítačovej sieti. Jedným z týchto spôsobov je TFTP, čo znamená Trivial File Transfer Protocol. TFTP nevyžaduje autentizáciu a používa UDP protokol.

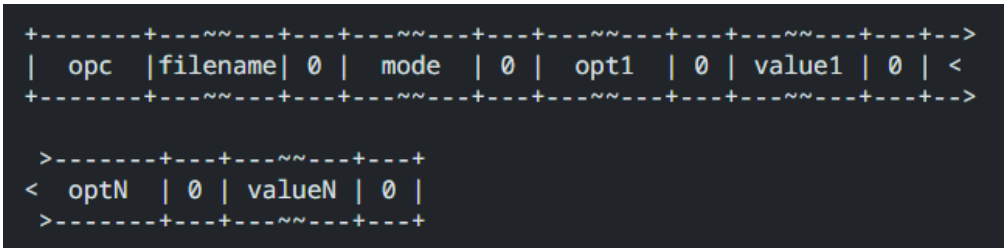
Všeobecné vlastnosti

[1] Akýkoľvek prenos začína požiadavkou na čítanie (Read Request, ďalej len RRQ) alebo zápis (Write Request, ďalej len WRQ) súboru, ktorý slúži aj na vyžiadanie spojenia. Názov súboru (Filename) značí, do akého súboru sa majú data uložiť, alebo z ktorého súbor sa majú data čítať. Mód (Mode) obsahuje formát dát. “netascii” alebo “octet”. Opkód pre RRQ je 1 a pre WRQ 2. Klient alebo server môže po určitom čase timeoutnúť a preposlať posledný poslaný paket.



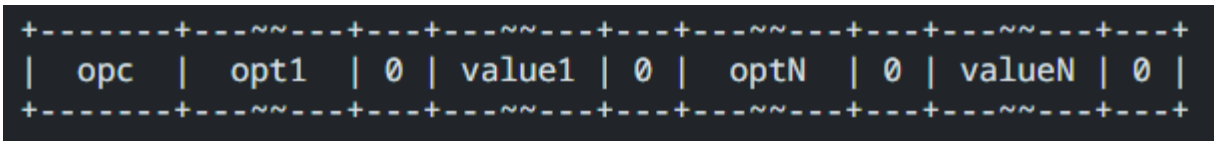
WRQ/RRQ paket

V prípade rozšírenia s možnosťami (Option Extension) [2], WRQ/RRQ paket obsahuje ešte dvojice (možnosť, hodnota). Tento paket sa považuje za žiadosť klienta o vyjednanie možností. Maximálna veľkosť WRQ/RRQ paketu je 512 oktetov.



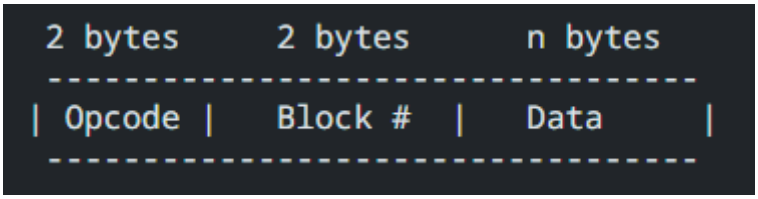
WRQ/RRQ paket s Možnosťami

Server po prijatí WRQ/RRQ paketu posiela paket potvrdenie možností (Option acknowledgment, ďalej len OACK), ktorý sa používa na potvrdenie žiadosti klienta o vyjednaní možností. Chybový paket s kódom 8 hlási termináciu spojenia, z dôvodu vyjednávania možností. Opkóde je 6.



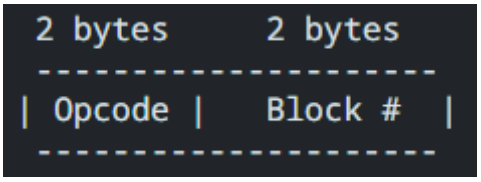
OACK paket

Paket ktorý obsahuje data sa nazýva data paket (Data packet, ďalej len DATA). Opkód je 3. Taktiež obsahuje číslo bloku dát a samotné data. Veľkosť dát menej ako 512B, alebo predom dohodnutej veľkosti pri vyjednávaní značí koniec prenosu.



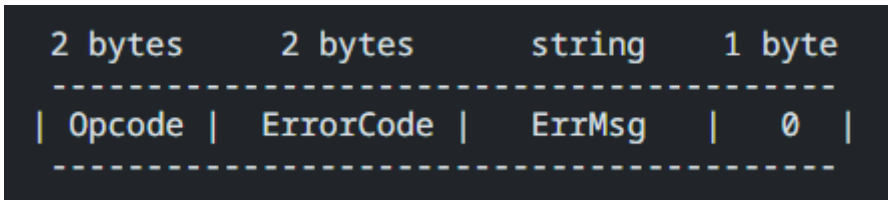
DATA paket

Každý DATA paket je nutné potvrdiť tzv. potvrdzovací paket (acknowledgement packet, ďalej len ACK). Tento paket značí, že príjemca dostal data s určitým číslom bloku. Opkód je 4.



ACK paket

Chybový paket (Error packet, ďalej len ERROR) môže byť potvrdením akéhokoľvek iného typu paketu. Opkód je 5. Chybový kód (ErrorCode) udáva povahu chyby. Všetky chybové kódy okrem 5 hlásia koniec spojenia. Chybové hlásenie (ErrMsg) je určené pre uživateľa a mala by byť v netascii.



ERROR paket

Základné chybové kódy sú dolu na obrázku. [2] Kód 8 hlási hlási termináciu spojenia, z dôvodu vyjednávania možností.

Value	Meaning
0	Not defined, see error message (if any).
1	File not found.
2	Access violation.
3	Disk full or allocation exceeded.
4	Illegal TFTP operation.
5	Unknown transfer ID.
6	File already exists.
7	No such user.

[1] Chybové kódu ERROR paketu

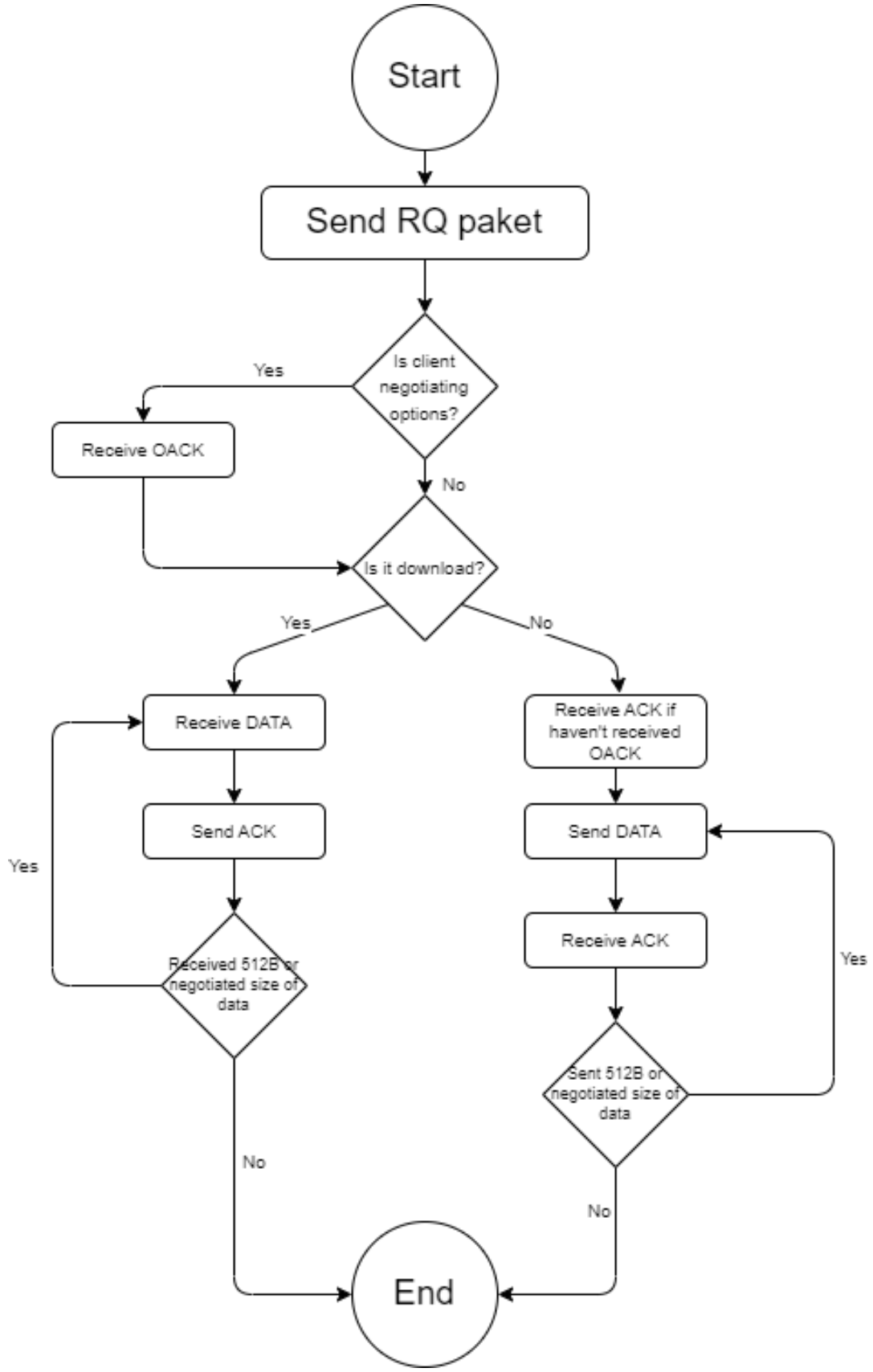
Download

Ak bol posledný prijatý paket klientom OACK, klient musí potvrdiť prijatie OACK a poslať ACK paket s číslom bloku 0. Ak klient nevyjednával, rovno čaká na dáta. Server začne posielať DATA, na ktoré musí príjemca odpovedať ACK paketom s daným číslom prijatého bloku dát.

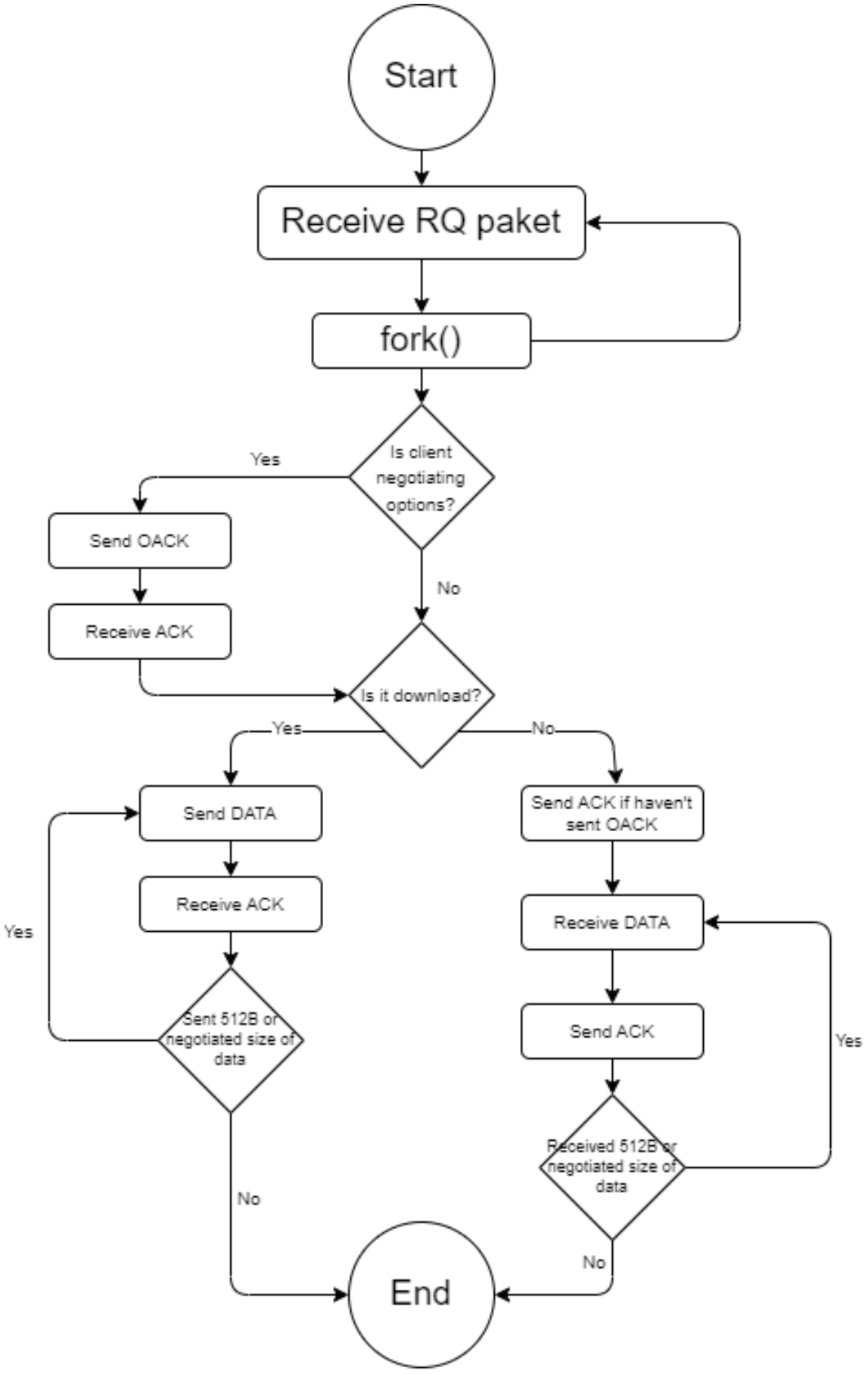
Upload

Ak kleint vyjednáva možnosti, tak očakáva OACK paket, po ktorom začne klient posielať DATA pakety. Ak na druhú stranu klient nevyjednával, musí od servera namiesto OACK dostať ACK paket s blokom 0. Následne môže posielať DATA pakety.

2. Návrh aplikácie



Návrh klienta



Návrh serveru

3. Popis implementácie

Klient

Klient je implementovaný v súboroch tftp-client.c a tftp-client.h. V maine spracujem command line argumenty, vytvorím soket, a nakonfigurujem adresu serveru. Následne sa podľa získaných argumentov v ife rozhodne či ide o download alebo upload.

V prípade dowloadu posielam RRQ paket s prípadnými možnosťami (tie sa dajú nastaviť na začiatku mainu). Následne otváram súbor na čítanie. Ak klient poslal aj možnosti očakávam OACK a posielam ACK. Premenná block identifikuje ktorý block je aktuálne spracovaný. Všetky parametre paketov, ktoré majú veľkosť 2B používajú uint_16t Tento parameter posielam do sendAckPacket() a receiveDataPacket() aby sa vedelo, ktorý ACK sa posiela a ktorý DATA paket sa očakáva. Toto chovanie je implementované pomocou do while cyklu, ktorý kontroluje, či klient prijal očakávaný počet dát (512B, alebo inú vyjednanú hodnotu). Po prijatí menej dát ako bolo očakávané končí prenos súboru.

V prípade upload načítam data a stdinu do dynamicky alokvoaného priestoru, ktorý sa postupne podľa potreby realokuj. Náslende posielam WRQ paket s prípadnými možnosťami (tie sa dajú nastaviť na začiatku mainu). Ak klient poslal aj možnosti očakávam OACK, inak očakávam ACK. Po obdržaní tohto paketu posielam v do while cykle postupne data pomocou funkcie sendDataPacket() a očakávam ACK paket pomocou funkcie receiveAckPacket. Tu je taktiež použitá premenná block, ktorá identifikuje, ktorý blok je práve spracovaný. Do while cyklus sa skončí keď klient pošle menej dát ako 512B, alebo menej ako vyjednaná hodnota.

Server

Server je implementovaný v súboroch tftp-server.c a tftp-server.h. V maine spracujem command line argumenty, vytvorím hlavný soket, nakonfigurujem adresu a nabindujem ju aby počúvala na určitom porte. Následne je hlavný while(true) loop ktorý čaká na prichádzajúce WRQ/RRQ pakety. Keď príde paket, program sa forkne. Child process vytvorí nový soket a hlavný process pokračuje počúvať ďalšie WRQ/RRQ pakety. Ak prišiel paket s vyjednávaním, posielam OACK so schválenými options. Následne sa podľa bool send_file, ktorý sa nastavil vo funkcii receiveRqPacket rozhodne či ide o download alebo upload.

V prípade downloadu otvorím súbor na čítanie. Ak prebieha vyjednávanie očakávam ACK packet. Následne sa pomocou do while cyklu posielajú DATA pakety s využitím sendDataPacket() funkcie. Premenná block zase indikuje, ktorý blok dát je aktuálne spracovaný. Ak server pošle menej dát, cyklus končí.

V prípade uploadu otváram súbor na zápis. Ak sa nevyjednávajú options server posiela ACK paket. Potom v do while cykle dostáva DATA pakety, ktoré postupne zapisuje do súboru. Ak príde menej dát ako je očakávané, cyklus končí.

Server konvertuje dáta podľa módu (netascii/octet) v sendDataPacket() a receiveDataPacket() funkciách.

Spoločné

Klient aj Server program je ošetrený voči chybnej komunikácii príslušnými ERROR paketami. Na konci funkii receive/send data/ack paket volám funkcie na printovanie paketov na stderr. Na stdout sú printované lokálne errory.Vždy keď klient alebo server čaká na paket je volaná funkcia handleTimeout(), ktorá podľa premennej timeout čaká, či prichádza páket. Ak klient alebo server timeoutne, je znova poslaný posledný paket. Nenašiel som počet retransmitov a tak som dal 3.

4. Návod na použitie

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

- -h IP adresa/doménový názov vzdialeného serveru
- -p port vzdialeného serveru
    - pokiaľ nie je špecifikovaný predpokladá sa východzí podľa špecifikácie
- -f cesta k sťahovanému súboru na serveru (download)
    - pokiaľ nie je špecifikovaný používá sa obsah stdin (upload)
- -t cesta, pod ktorou bude súbor na vzdialenom serveru/lokálne uložený

tftp-server [-p port] root_dirpath

- -p miestny port, na ktorom bude server očakávať príchodzie spojenie
- cesta k adresári, pod ktorým sa budú ukládať príchodzie súbory

Program postupne vypysuje na štandardný chyboý výstup (stderr) správy v následujúcom formáte:

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} potom vo formáte podľa poradia v dátovom prenose:

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

Lokálne chybové hlásenia sú zobrazované na štandardný výstup (stdout)

5. Bibliografia

[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.