VYSOKÉ UČENÍ TECHNICKÉ V BRNĚ
BRNO UNIVERSITY OF TECHNOLOGY

FAKULTA INFORMAČNÍCH TECHNOLOGIÍ
FACULTY OF INFORMATION TECHNOLOGY

ÚSTAV INFORMAČNÍCH SYSTÉMŮ
DEPARTMENT OF INFORMATION SYSTEMS

ANALÝZA FTP

FTP ANALYSIS

ODBORNÁ PRÁCE

AUTOR PRÁCE — **AUGUSTIN MACHYŇÁK**
AUTHOR

CVIČÍCÍ — **Mgr. Ing. PAVEL OČENÁŠEK, Ph.D.**
INSTRUCTOR

BRNO 2022

Abstrakt

Tato práce se zabývá analýzou protokolu FTP.

Abstract

This work deals with the analysis of the FTP protocol.

Klíčová slova

FTP, analýza, wireshark

Keywords

FTP, analýza, wireshark

Citace

MACHYŇÁK, Augustin. *Analýza FTP*. Brno, 2022. Odborná práce. Vysoké učení technické v Brně, Fakulta informačních technologií. Cvičící Mgr. Ing. Pavel Očenášek, Ph.D.

Analýza FTP

Prohlášení

Prohlašuji, že jsem tuto odbornou práci vypracoval samostatně. Uvedl jsem všechny literární prameny, publikace a další zdroje, ze kterých jsem čerpal.

.....
Augustin Machyňák
28. dubna 2022

Obsah

Kapitola 1

Úvod

Tato práce se zabývá analýzou FTP (File Transfer Protocol) - protokol sloužící pro přenos souborů mezi počítači v síti definován v RFC 959 [1].

Na FTP server a klienta je v této práci nahlíženo jako *černé skříňky* a pomocí nástroje wireshark je pozorováno, jak probíhá jejich komunikace.

Pro realizaci jsou nutné následující aplikace:

- FTP server
- FTP klient
- netcat
- Aplikaci pro analýzu příchozích a odchozích paketů (Wireshark)

FTP klient je dostupný na většině operačních systémech. Pro FTP server je možné použít vsftpd (Linux) nebo IIS Manager (Windows 10). Wireshark je možné stáhnout z oficiálních stránek.

Tato práce používá vsftpd pro FTP server, FTP klienta dostupného na (snad) všech známých linuxových distribucích a wireshark verze 3.2.3. Komunikace probíhá na rozhraní loopback (localhost). Ve wiresharku je zvoleno rozhraní loopback a filtr *ftp or ftp-data*. Pro účely této práce byl vytvořen uživatel *test_user* s heslem *test*, přes kterého je možné se připojit na FTP server.

Kapitola 2

Analýza komunikace

Prvně je spuštěn FTP server pomocí

```
$ sudo service vsftpd start
```

Po spuštění FTP serveru je proveden pokus o připojení klienta:

```
$ ftp localhost
```

Klient byl připojen k serveru a server zaslal následující odpověď: "220 (vsFTPd 3.0.3)". První část odpovědi zřejmě značí kód odpovědi. To je možné potvrdit v programu wireshark, který tuto zprávu rozdělil na 2 části a ukazuje následující:

- Response code: Service ready for new user (220)
- Response arg: (vsFTPd 3.0.3)

Pokud by cílem bylo nalezení zranitelností, tak jsme již získali užitečnou informaci - jméno serveru. Avšak jediná známá zranitelnost pro tuto verzi serveru je dle exploit-db DoS útok, takže nic zajímavého není možné provést.

Z této odpovědi lze předpokládat (což následující komunikace i potvrdí), že se odpovědi skládají ze 2 částí - kódu, který říká, jak server zareagoval na požadavek klienta, a argumentu, který může být brán jako odpověď či požadavek pro klienta.

Aplikace klienta nyní žádá o zadání jména a hesla uživatele. Při zadání jména je zaslán serveru příkaz *USER [vstup]* a odpovědí je "331 Please specify the password." (i když je zadáno neexistující jméno). Při následném zadání hesla je zaslán serveru příkaz *PASS [vstup]* a odpovědí je buď "530 Login incorrect.", pokud bylo zadáno správně jméno a heslo, nebo "Login successful.", pokud je jméno nebo heslo chybné. Zřejmě z těchto odpovědí není ani možné poznat, jestli uživatel s daným uživatelským jménem vůbec existuje.

Zde je možné vyzkoušet, jak server bude reagovat na různé příkazy. Pomocí aplikace netcat a příkazu

```
$ netcat localhost 21
```

je provedeno připojení na server. Server opět zaslal odpověď "220 (vsFTPd 3.0.3)". Pokud vyzkoušíme zaslat příkaz *PASS test*, tak server odpoví "503 Login with USER first.". Reakce serveru na jakýkoliv jiný příkaz je "530 Please login with USER and PASS.". Pokud není zadán žádný příkaz, server po několika minutách odpoví s "421 Timeout.". a ukončí spojení. Pokud je zaslán příkaz "USER test_user" před odpovědí s kódem 220 od serveru, tak je

sice získána očekáváná odpověď s kódem 331, avšak server v této chvíli "zamrzne"- přestane odpovídat na jakékoliv příkazy od klienta a jediná možnost je ukončit spojení ze strany klienta nebo počkat na ukončení spojení ze strany serveru.

Zaslání zprávy, která je delší než 4096 znaků, vede k odpovědi “500 Input line too long.” a server následně ukončí spojení.

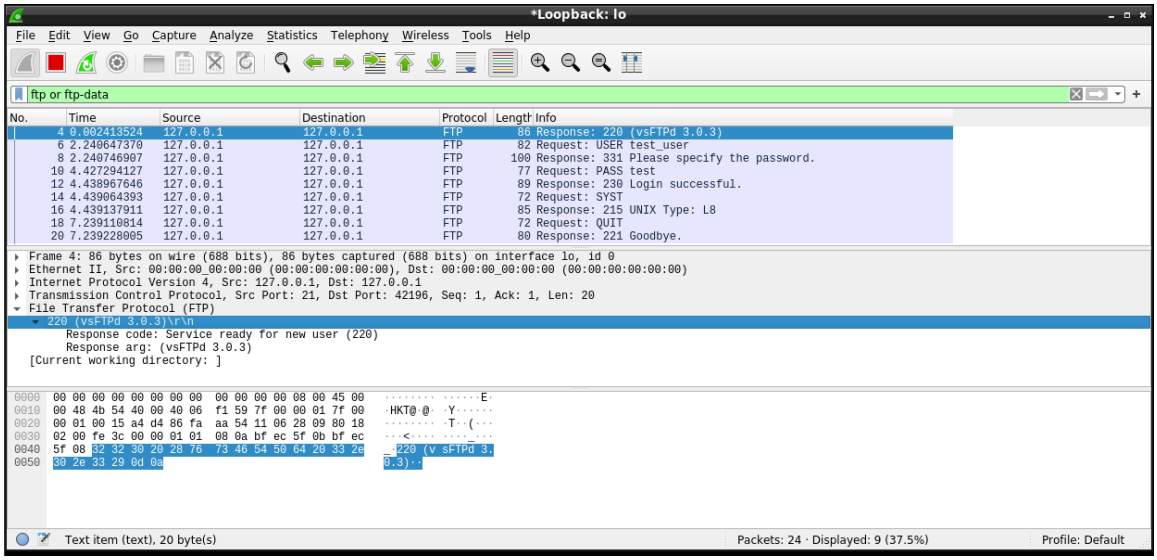
Po úspěšném připojení a autentizaci (“USER test_user” a “PASS test”) je již možné zaslat větší množství příkazů. Seznam těchto příkazů je možné získat ze serveru pomocí příkazu *HELP*. Server na tento příkaz reaguje odpovědí “214-The following commands are recognized.” a následně zašle 4 zprávy s příkazy. Každý příkaz je oddělen mezerou, obsahuje pouze velká písmena, není delší než 4 znaky a každá zpráva není delší než 72 znaků. Reakce serveru na příkaz *HELP* vypadá následovně:

1. 214-The following commands are recognized.
2. ABOR ACCT ALLO APPE CDUP CWD DELE EPRT EPSV FEAT HELP LIST MDTM MKD
3. MODE NLST NOOP OPTS PASS PASV PORT PWD QUIT REIN REST RETR RMD RNFR
4. RNTO SITE SIZE SMNT STAT STOR STOU STRU SYST TYPE USER XCUP XCWD XMKD
5. XPWD XRMD
6. 214 Help OK.

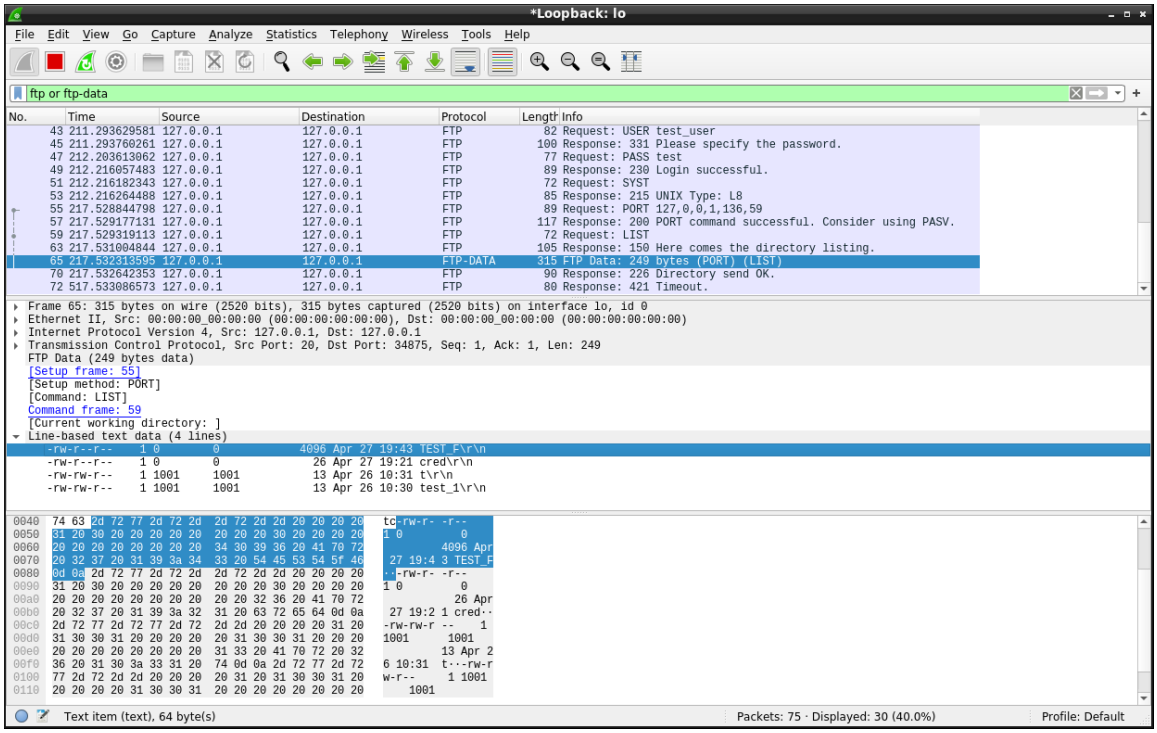
Veškerá komunikace probíhala doposud na portu 21. FTP však používá ještě jeden port - port 20 - pro přenos dat. Pro účely demonstrace bude analyzován ještě jeden příkaz - LIST. Tento příkaz slouží k vypsání obsahu současného adresáře.

Pokud je po autentizaci (pomocí netcatu) zadán příkaz LIST, tak je obdržena zpráva “425 Use PORT or PASV first.”. Pro příkazy pro přenos dat (mezi které patří i příkaz *LIST*) používá FTP port 20 a je nutné pomocí příkazu *PORT* sdělit serveru, na kterou adresu a port má zasílat tyto data. FTP klient obvykle tuto část dělá automaticky a to při zaslání některého z příkazu, které posílají data na jiný port. Při použití příkazu *ls* (LIST) v klientské aplikaci se provede následující:

1. Klient začne naslouchat na libovolném portu
2. Klient zašle příkaz PORT, jehož argumenty identifikují adresu a port, na kterém naslouchá
3. Pokud nedošlo k žádné chybě na straně serveru, tak server zašle zpátky odpověď s kódem 200
4. Klient zašle příkaz LIST
5. Server se připojí na adresu a port, který byl zaslán minulým PORT příkazem
6. Server zašle data a ukončí spojení



Obrázek 2.1: Komunikace mezi klientem a serverem zachycena programem wireshark. Klient se autentizuje a následně odpojí od serveru.



Obrázek 2.2: Komunikace mezi klientem a serverem zachycena programem wireshark. Klient se autentizuje, zašle serveru informace o adrese a portu na kterém naslouchá (příkaz PORT) a po zaslání příkazu LIST server odpovídá s daty na zadanou adresu a port. Po 5 minutách nečinnosti server ukončuje spojení (Timeout).

Kapitola 3

Zhodnocení

Pomocí nástroje wireshark bylo zjištěno, jak probíhá autentizace mezi klientem a serverem využívající FTP a jak FTP využívá datový port (20) pro přenos dat. Zřejmě bez znalosti protokolu, přes který klient a server funguje, a s použitím selského rozumu (například vyzkoušení příkazu HELP) je možné zjistit dost informací. Odpověď serveru obvykle obsahuje i jeho jasné požadavky, což není pravidlem pro všechny protokoly. Tento fakt velmi ulehčuje analýzu.

Obtížné může být například zjištění sémantiky příkazu PORT. Příkaz PORT vypadá následovně

```
PORT <n1>, <n2>, <n3>, <n4>, <p1>, <p2>
```

kde *n1, n2, n3, n4* představuje 4 čísla v ip adrese hosta a *p1 a p2* slouží pro výpočet portu. Port se z *p1 a p2* vypočítá pomocí vzorce $port = (p1 * 256) + p2$ což může být velmi neintuitivní.

Byly také zjištěny některé limity, které však můžou být specifické pro konkrétní implementaci (maximální délka zprávy 4096 znaků) a otestovány některé krajní případy (co se stane, když klient začne komunikovat před první zprávou od serveru ...).

Literatura

[1] Postel, J. a Reynolds, J. *RFC 959 - File Transfer Protocol (FTP)* [online]. 1985 [cit. 2022-04-28]. Dostupné z: https://datatracker.ietf.org/doc/html/rfc959.