

# Project 1

شیرین بهنامی نیا – نرگس منتظری

## Questions:

1- چرا این روند برعکس نیست یعنی کلاینت يك پورت را باز نمی کند و آن را به سرور اطلاع دهد و سرور فایل را روی آن پورت برای کلاینت ارسال کند؟

به طور کلی پروتکل ftp دو مود متفاوت دارد: active mode و passive mode . در passive mode که ما هم در حقیقت همون رو پیاده سازی کردیم دیتاپورت رندوم از سمت سرور تعیین میشود و برای کلاینت فرستاده میشود تا به آن کانکت شود اما در active mode دقیقاً برعکس این اتفاق میفتد. اگر چیزی به نام firewall نداشتیم عملاً این حالت برامون مشکل خاصی ایجاد نمیکرد اما در حال حاضر شرایط فرق داره. کلاینت ها عموماً پشت یک فایروال یا یک NAT قرار دارند و به همین دلیل فقط از طریق پورت های خاصی ( که بهش predefined میگن) از بیرون قابل دسترسی اند. پس وقتی سرور تلاش میکنه از طریق پورت رندومی که کلاینت براش فرستاده با اون ارتباط برقرار کنه به احتمال خیلی زیاد اون پورت جزو پورت های predefined نیست و در نتیجه توسط فایروال بلاک میشه و هیچ کانکشنی برقرار نمیشه. به همین دلیل default mode در این پروتکل passive هست.

2- نام این حمله چیست؟

اسم این حمله Directory Traversal Attack است.

زمانی رخ می دهد که حمله کننده بتواند به فراتر از فهرست های اصلی سرور و فایل هایی که اجازه دارد، دسترسی پیدا کند. اگر او چنین دسترسی را داشته باشد می تواند هم فایل هایی که مجاز نیست را دریافت کند و به اطلاعات آن ها دسترسی پیدا کند و هم ممکن است دستورات سیستم عامل را اجرا کند که باعث ایجاد مشکل در سرور می شود.

3- خروجی های سرور و کلاینت:

از خروجی های کلاینت جلوتر اسکرین شات قرار دادیم.

سرور:

23:05:44.501475

SUCCESS

Client1: CD command DONE.

23:05:49.586766

SUCCESS

Client1: PWD command DONE.

23:05:55.447418

SUCCESS

Client1: LIST command DONE.

23:06:00.924642

SUCCESS

Client1: CD command DONE.

23:06:08.076401

SUCCESS

Client1: Client connected to data channel:('127.0.0.1', 53207)

23:06:08.099984

SUCCESS

Client1: ['FTP.png'] uploaded.

23:09:37.380096

SUCCESS

Client1: Client connected to data channel:('127.0.0.1', 51925)

23:09:37.384971

ERROR

Client1: ['fileeeeeee.png'] not found.

## Wireshark:

1- handshaking packets

| tcp  |           |           |             |          |        |  |
|--|-----------|-----------|-------------|----------|--------|--|
| No.  | Time      | Source    | Destination | Protocol | Length | Info   |
| 7  | 23.316904 | 127.0.0.1 | 127.0.0.1   | TCP      | 56     | 58778 → 2121 [SYN] Seq=0 Win=65535 Len=0 MSS=65495 WS=256 SACK_PERM=1            |
| 8  | 23.317004 | 127.0.0.1 | 127.0.0.1   | TCP      | 56     | 2121 → 58778 [SYN, ACK] Seq=0 Ack=1 Win=65535 Len=0 MSS=65495 WS=256 SACK_PERM=1 |
| 9  | 23.317203 | 127.0.0.1 | 127.0.0.1   | TCP      | 44     | 58778 → 2121 [ACK] Seq=1 Ack=1 Win=2619648 Len=0                                 |
| <div> <div>Packet comments</div> <div> <div>handshaking</div> <div> <div>Frame 9: 44 bytes on wire (352 bits), 44 bytes captured (352 bits) on interface \Device\NPF_{Loopback, id 0}</div> <div>Null/Loopback</div> <div>Internet Protocol Version 4, Src: 127.0.0.1, Dst: 127.0.0.1</div> <div>Transmission Control Protocol, Src Port: 58778, Dst Port: 2121, Seq: 1, Ack: 1, Len: 0</div> </div> </div> </div> |           |           |             |          |        |  |

-Step1(SYN): Client to Server (informs the server that the client is likely to start communicate.)

-Step2(SYN-ACK): Server to Client (signifies the response of the segment it received and SYN signifies with what sequence number it is likely to start the segments with)

-Step3(ACK): Client to Server (client acknowledges the response of the server)

## 2-TCP Packet size limitation

| No. | Time       | Source    | Destination | Protocol | Length | Info  |
|-----|------------|-----------|-------------|----------|--------|---|
| 26  | 85.633172  | 127.0.0.1 | 127.0.0.1   | TCP      | 44     | 59880 → 2121 [ACK] Seq=15 Ack=53 Win=2619648 Len=0                                |
| 27  | 117.603066 | 127.0.0.1 | 127.0.0.1   | TCP      | 60     | 59880 → 2121 [PSH, ACK] Seq=15 Ack=53 Win=2619648 Len=16                          |
| 28  | 117.603157 | 127.0.0.1 | 127.0.0.1   | TCP      | 44     | 2121 → 59880 [ACK] Seq=53 Ack=31 Win=2619648 Len=0                                |
| 29  | 117.603266 | 127.0.0.1 | 127.0.0.1   | TCP      | 49     | 2121 → 59880 [PSH, ACK] Seq=53 Ack=31 Win=2619648 Len=5                           |
| 30  | 117.603354 | 127.0.0.1 | 127.0.0.1   | TCP      | 44     | 59880 → 2121 [ACK] Seq=31 Ack=58 Win=2619648 Len=0                                |
| 31  | 117.603686 | 127.0.0.1 | 127.0.0.1   | TCP      | 56     | 51630 → 24653 [SYN] Seq=0 Win=65535 Len=0 MSS=65495 WS=256 SACK_PERM=1            |
| 32  | 117.603877 | 127.0.0.1 | 127.0.0.1   | TCP      | 56     | 24653 → 51630 [SYN, ACK] Seq=0 Ack=1 Win=65535 Len=0 MSS=65495 WS=256 SACK_PERM=1 |
| 33  | 117.604068 | 127.0.0.1 | 127.0.0.1   | TCP      | 44     | 51630 → 24653 [ACK] Seq=1 Ack=1 Win=2619648 Len=0                                 |
| 34  | 117.605908 | 127.0.0.1 | 127.0.0.1   | TCP      | 47     | 24653 → 51630 [PSH, ACK] Seq=1 Ack=1 Win=2619648 Len=3                            |
| 35  | 117.606007 | 127.0.0.1 | 127.0.0.1   | TCP      | 44     | 51630 → 24653 [ACK] Seq=1 Ack=4 Win=2619648 Len=0                                 |
| 36  | 117.603826 | 127.0.0.1 | 127.0.0.1   | TCP      | 51     | 24653 → 51630 [PSH, ACK] Seq=4 Ack=1 Win=2619648 Len=7                            |
| 37  | 117.604005 | 127.0.0.1 | 127.0.0.1   | TCP      | 44     | 51630 → 24653 [ACK] Seq=1 Ack=11 Win=2619648 Len=0                                |
| 38  | 118.199089 | 127.0.0.1 | 127.0.0.1   | TCP      | 65539  | 24653 → 51630 [ACK] Seq=11 Ack=1 Win=2619648 Len=65495                            |
| 39  | 118.199428 | 127.0.0.1 | 127.0.0.1   | TCP      | 65539  | 24653 → 51630 [ACK] Seq=65586 Ack=1 Win=2619648 Len=65495                         |
| 40  | 118.199710 | 127.0.0.1 | 127.0.0.1   | TCP      | 65539  | 24653 → 51630 [ACK] Seq=131001 Ack=1 Win=2619648 Len=65495                        |
| 41  | 118.199949 | 127.0.0.1 | 127.0.0.1   | TCP      | 65539  | 24653 → 51630 [ACK] Seq=196496 Ack=1 Win=2619648 Len=65495                        |
| 42  | 118.200196 | 127.0.0.1 | 127.0.0.1   | TCP      | 65539  | 24653 → 51630 [ACK] Seq=261991 Ack=1 Win=2619648 Len=65495                        |
| 43  | 118.200535 | 127.0.0.1 | 127.0.0.1   | TCP      | 65539  | 24653 → 51630 [ACK] Seq=327486 Ack=1 Win=2619648 Len=65495                        |
| 44  | 118.200873 | 127.0.0.1 | 127.0.0.1   | TCP      | 65539  | 24653 → 51630 [ACK] Seq=392981 Ack=1 Win=2619648 Len=65495                        |
| 45  | 118.201320 | 127.0.0.1 | 127.0.0.1   | TCP      | 65539  | 24653 → 51630 [ACK] Seq=458476 Ack=1 Win=2619648 Len=65495                        |
| 46  | 118.201586 | 127.0.0.1 | 127.0.0.1   | TCP      | 65539  | 24653 → 51630 [ACK] Seq=623971 Ack=1 Win=2619648 Len=65495                        |
| 47  | 118.201814 | 127.0.0.1 | 127.0.0.1   | TCP      | 65539  | 24653 → 51630 [ACK] Seq=589466 Ack=1 Win=2619648 Len=65495                        |
| 48  | 118.203440 | 127.0.0.1 | 127.0.0.1   | TCP      | 44     | 51630 → 24653 [ACK] Seq=1 Ack=654961 Win=2619648 Len=0                            |
| 49  | 118.203819 | 127.0.0.1 | 127.0.0.1   | TCP      | 65539  | 24653 → 51630 [ACK] Seq=654961 Ack=1 Win=2619648 Len=65495                        |
| 50  | 118.204207 | 127.0.0.1 | 127.0.0.1   | TCP      | 65539  | 24653 → 51630 [ACK] Seq=720456 Ack=1 Win=2619648 Len=65495                        |
| 51  | 118.204519 | 127.0.0.1 | 127.0.0.1   | TCP      | 65539  | 24653 → 51630 [ACK] Seq=785951 Ack=1 Win=2619648 Len=65495                        |
| 52  | 118.204808 | 127.0.0.1 | 127.0.0.1   | TCP      | 65539  | 24653 → 51630 [ACK] Seq=851446 Ack=1 Win=2619648 Len=65495                        |
| 53  | 118.205223 | 127.0.0.1 | 127.0.0.1   | TCP      | 65539  | 24653 → 51630 [ACK] Seq=916941 Ack=1 Win=2619648 Len=65495                        |
| 54  | 118.205562 | 127.0.0.1 | 127.0.0.1   | TCP      | 65539  | 24653 → 51630 [ACK] Seq=982436 Ack=1 Win=2619648 Len=65495                        |
| 55  | 118.205807 | 127.0.0.1 | 127.0.0.1   | TCP      | 700    | 24653 → 51630 [PSH, ACK] Seq=1047931 Ack=1 Win=2619648 Len=656                    |
| 56  | 118.206707 | 127.0.0.1 | 127.0.0.1   | TCP      | 44     | 51630 → 24653 [ACK] Seq=1 Ack=1048587 Win=2619648 Len=0                           |
| 57  | 118.207458 | 127.0.0.1 | 127.0.0.1   | TCP      | 44     | 24653 → 51630 [FIN, ACK] Seq=1048587 Ack=1 Win=2619648 Len=0                      |
| 58  | 118.207591 | 127.0.0.1 | 127.0.0.1   | TCP      | 44     | 51630 → 24653 [ACK] Seq=1 Ack=1048588 Win=2619648 Len=0                           |
| 59  | 118.210511 | 127.0.0.1 | 127.0.0.1   | TCP      | 44     | 51630 → 24653 [FIN, ACK] Seq=1 Ack=1048588 Win=2619648 Len=0                      |
| 60  | 118.210736 | 127.0.0.1 | 127.0.0.1   | TCP      | 44     | 24653 → 51630 [ACK] Seq=1048588 Ack=2 Win=2619648 Len=0                           |

-ماکزیم اندازه پکت در این پروتکل تقریباً برابر 64 کیلوبایت است. (با توجه به وایرشارک 65539 بایت به طور حدودی) این مقدار با توجه به سخت افزار مورد استفاده در شبکه تعیین میشود.(mtu)

MTU : MTU is the maximum size of the data transfer limit set by hardware in a network. The packet size should never exceed MTU.

این پروتکل برای ارسال دیتا وقتی با حجمی بیشتر از این مقدار مواجه میشود خودش دیتا را به بسته هایی با ماکزیم سایز 64k میشکند و در پکت های جداگانه ارسال میکند که در مقصد(کلاینت) دوباره به هم متصل میشن و دیتای اصلی رو تشکیل میدن. ( هر پکت با توجه به اینکه چه مسیری بهینه تره بر اش، مسیر متفاوتی رو طی میکنه.)  
به طور کلی استفاده از پکت های کوچکتر احتمال loss of data رو کم میکنه و network latency رو کم میکنه.(همون lag)  
در انتقال bigfile.bin باتوجه به حجم 1024 کیلوبایتی تقریباً همنطور که انتظار داشتیم دیتا در 16 تا پکت ارسال شده است.

Ngrok:

Part1:

```
C:\Windows\System32\cmd.exe - ngrok tcp 2121
ngrok
Session Status online
Account Shirin (Plan: Free)
Version 3.0.2
Region Europe (eu)
Latency 149.9905ms
Web Interface http://127.0.0.1:4040
Forwarding tcp://2.tcp.eu.ngrok.io:16560 -> localhost:2121
Connections
```

|   | t1 | opn | rt1          | rt5  | p50    | p90    |
|---|----|-----|--------------|------|--------|--------|
| 1 | 6  | 1   | 0.01         | 0.00 | 359.39 | 359.39 |
| 7 |    |     | PORT = 16560 |      |        |        |

19:16:08.535781

SUCCESS

Connected to 2.tcp.eu.ngrok.io : 16560

-----

Call one of the following functions:

|                 |                      |
|-----------------|----------------------|
| 1.HELP          | : Show this commands |
| 2.LIST          | : List files         |
| 3.PWD           | : Show current dir   |
| 4.CD dir_name   | : Change directory   |
| 5.DWLD dir_name | : Download file      |
| 6.QUIT          | : Exit               |

-----

-----  
Enter command: cd dir1

19:16:24.119584

SUCCESS

\dir1/  
-----

Call one of the following functions:

|                 |                      |
|-----------------|----------------------|
| 1.HELP          | : Show this commands |
| 2.LIST          | : List files         |
| 3.PWD           | : Show current dir   |
| 4.CD dir_name   | : Change directory   |
| 5.DWLD dir_name | : Download file      |
| 6.QUIT          | : Exit               |

-----

Enter command: cd fff

19:18:13.680403

ERROR

fff not found.  
-----

Call one of the following functions:

|                 |                      |
|-----------------|----------------------|
| 1.HELP          | : Show this commands |
| 2.LIST          | : List files         |
| 3.PWD           | : Show current dir   |
| 4.CD dir_name   | : Change directory   |
| 5.DWLD dir_name | : Download file      |
| 6.QUIT          | : Exit               |

-----

Enter command: pwd

19:17:11.788230

SUCCESS

\dir1/

-----

Call one of the following functions:

- 1.HELP : Show this commands
  - 2.LIST : List files
  - 3.PWD : Show current dir
  - 4.CD dir\_name : Change directory
  - 5.DWLD dir\_name : Download file
  - 6.QUIT : Exit
- 

Enter command: LIST

19:17:06.138089

SUCCESS

bigFile.bin

> inner

Total size: 1048576

-----

Call one of the following functions:

- 1.HELP : Show this commands
  - 2.LIST : List files
  - 3.PWD : Show current dir
  - 4.CD dir\_name : Change directory
  - 5.DWLD dir\_name : Download file
  - 6.QUIT : Exit
-

```

Enter command: help

-----

Call one of the following functions:
1.HELP           : Show this commands
2.LIST           : List files
3.PWD           : Show current dir
4.CD dir_name    : Change directory
5.DWLD dir_name  : Download file
6.QUIT          : Exit

-----

```

## Part2:

|    |           |           |           |     |   |
|----|-----------|-----------|-----------|-----|---|
| 17 | 12.831859 | :::1      | :::1      | TCP | 76 60499 → 2121 [SYN] Seq=0 Win=65535 Len=0 MSS=65475 WS=256 SACK_PERM=1            |
| 18 | 12.831951 | :::1      | :::1      | TCP | 64 2121 → 60499 [RST, ACK] Seq=1 Ack=1 Win=0 Len=0                                  |
| 19 | 13.146714 | 127.0.0.1 | 127.0.0.1 | TCP | 56 60500 → 2121 [SYN] Seq=0 Win=65535 Len=0 MSS=65495 WS=256 SACK_PERM=1            |
| 20 | 13.146987 | 127.0.0.1 | 127.0.0.1 | TCP | 56 2121 → 60500 [SYN, ACK] Seq=0 Ack=1 Win=65535 Len=0 MSS=65495 WS=256 SACK_PERM=1 |
| 21 | 13.147222 | 127.0.0.1 | 127.0.0.1 | TCP | 44 60500 → 2121 [ACK] Seq=1 Ack=1 Win=2619648 Len=0                                 |
| 22 | 13.152804 | 127.0.0.1 | 127.0.0.1 | TCP | 46 2121 → 60500 [PSH, ACK] Seq=1 Ack=1 Win=2619648 Len=2                            |
| 23 | 13.152909 | 127.0.0.1 | 127.0.0.1 | TCP | 44 60500 → 2121 [ACK] Seq=1 Ack=3 Win=2619648 Len=0                                 |
| 24 | 28.162419 | 127.0.0.1 | 127.0.0.1 | TCP | 45 [TCP Keep-Alive] 60500 → 2121 [ACK] Seq=0 Ack=3 Win=2619648 Len=1                |

سه تا بسته ی اول مربوط به handshaking است. ( دو تا بسته ی پایین تر که فلگ psh دارند در حالت قبلی هم مشاهده میشد.) اینجا حالت loopback است و تفاوت قابل ملاحظه ای با حالت قبلی ندارد. اما به طور کلی ( هرچند من فقط موفق شدم در حالت loopback پکت ها را ببینم) اگر بتوانیم پکت ها را بیرون از حالت loopback ببینیم، پورتهی که سرور مشخص کرده هم در قسمت info مشخص است و منطقاً درس مبدا و مقصد هم متفاوت است.

یه تفاوتی که اینجا به چشم میاد نسبت به حالت قبلی اینه که اینجا تعداد زیادی بسته TCP-Keep-Alive داریم که وقتی دستوری وارد نمی کنیم فرستاده میشن که مطمئن بشن یک کانکشن هنوز فعاله.

## Part3:

مشکلی که وجود دارد این است که پورت دیتا چنل به صورت رندوم تولید میشه و ما نمیتونیم حین اجرا به شیوه فعلی برای اون ip عمومی بسازیم.

ولی این کار ممکن است. به کمک کتابخانه pyngrok در پایتون می توان ابتدا شماره پورت رندومی را ایجاد کرد و سپس یک کانکشن با ngrok برقرار کرد و آدرس هاست و شماره پورت آن را برای کلاینت ارسال کرد. سپس کلاینت به آدرسی که دریافت می کند وصل و فایل مورد نظر، از طریق آن کانال از سرور ارسال می شود .

البته نسخه رایگان ngrok مشکلاتی مانند محدودیت زمان و پهنای باند دارد که برای ارسال فایل های بزرگ از سرور به کلاینت مشکل ساز است و نمی توان فایل را یکجا ارسال کرد بلکه باید فایل را به تکه های کوچکتری تقسیم بندی کرد.

( ما این کار را در فایل جداگانه ای به اسم use-ngrok انجام داده ایم و آن را در کنار فایل اصلی قرار دادیم)