tcpdump примеры

ПРЕДСТАВЛЕНИЕ НЕОБРАБОТАНОГО ВЫВОДА

Подробный вывод без разрешения имен хостов или номеров портов, абсолютных порядковых номеров и удобочитаемых временных меток.

```
# tcpdump -ttttnnvvS
```

НАЙДИТЕ ТРАФИК ПО IP

Один из самых распространенных запросов, это покажет вам трафик из 1.2.3.4, будь то источник или место назначения.

```
# tcpdump host 1.2.3.4
```

ПОСМОТРЕТЬ БОЛЬШЕ ИНФОРМАЦИИ О ПАКЕТЕ С ВЫВОДОМ НА ШЕСТНАДЦАТЕРИЧНУЮ СИСТЕМУ

Шестнадцатеричный вывод полезен, когда вы хотите увидеть содержимое рассматриваемых пакетов, и его часто лучше всего использовать, когда вы изолируете несколько кандидатов для более тщательного изучения.

```
# tcpdump -nnvXSs 0 -c1 icmp
```

```
tcpdump: listening on eth0, link-type EN10MB (Ethernet),
23:11:10.370321 IP
(tos 0x20, ttl 48, id 34859, offset 0, flags [none], length:
84)
69.254.213.43 > 72.21.34.42: icmp 64: echo request seq 0
0x0000:  4520  0054  882b  0000  3001  7cf5  45fe  d52b
E..T.+..0.|.E..+
0x0010: 4815 222a 0800 3530 272a 0000 25ff d744 H."..50'..%..D
0x0020:  ae5e  0500  0809  0a0b  0c0d  0e0f  1011  1213
.^..............
0x0030:  1415  1617  1819  1a1b  1c1d  1e1f  2021  2223
.............!"#
0x0040: 2425 2627 2829 2a2b 2c2d 2e2f 3031 3233 $%&'()*+,-
```

```
./0123
0x0050: 3435 3637 4567
1 packets captured
1 packets received by filter
0 packets dropped by kernel
```

ФИЛЬТРАЦИЯ ПО ИСТОЧНИКАМ И НАЗНАЧЕНИЮ

Выделить трафик на основе источника или назначения очень просто, используя src и dst.

```
# tcpdump src 2.3.4.5
# tcpdump dst 3.4.5.6
```

ПОИСКОВЫЕ ПАКЕТЫ ПО СЕТИ

Чтобы найти пакеты, идущие в или из определенной сети, используйте опцию net. Вы можете комбинировать это с опциями src или dst.

```
# tcpdump net 1.2.3.0/24
```

ПОКАЗАТЬ ТРАФИК СВЯЗАНЫЙ СО СПЕЦИАЛЬНЫМ ПОРТОМ

Вы можете найти определенный порт трафика, используя опцию port, за которой следует номер порта.

```
# tcpdump port 3389
# tcpdump src port 1025
```

ПОКАЗАТЬ ТРАФИК ОДНОГО ПРОТОКОЛА

Если вы ищете определенный тип трафика, вы можете использовать tcp, udp, icmp и многие другие.

```
# tcpdump icmp
```

ПОКАЗАТЬ ТОЛЬКО ТРАФИК IP6

Вы также можете найти весь трафик IP6, используя опцию протокола.

```
# tcpdump ip6
```

НАЙДИТЕ ТРАФИК С ИСПОЛЬЗОВАНИЕМ ПОРТОВЫХ ДИАПАЗОНОВ

Вы также можете использовать диапазон портов, чтобы найти трафик.

```
# tcpdump portrange 21-23
```

НАЙДИТЕ ТРАФИК НА ОСНОВЕ РАЗМЕРА ПАКЕТА

Если вы ищете пакеты определенного размера, вы можете использовать эти параметры. Вы можете использовать маленький, большой или их соответствующие символы, которые вы ожидаете от математики.

```
# tcpdump less 32
# tcpdump greater 64
# tcpdump <= 128
```

ПИСЬМЕННЫЕ ЗАПИСИ В ФАЙЛ

Часто полезно сохранять результат пакетов в файл для анализа в будущем. Эти файлы известны как файлы PCAP (PEE-cap), и их можно обрабатывать сотнями различных приложений, включая сетевые анализаторы, системы обнаружения вторжений и, конечно же, самим tcpdump. Здесь мы пишем файл с именем capture_file, используя ключ -w.

```
# tcpdump port 80 -w capture_file
```

ЧТЕНИЕ ФАЙЛОВ PCAP

Вы можете читать файлы PCAP с помощью ключа -r. Обратите

внимание, что вы можете использовать все регулярные команды в tcpdump при чтении в файле; вы ограничены только тем фактом, что вы не можете захватывать и обрабатывать то, чего не существует в файле.

```
# tcpdump -r capture_file
```

РАСШИРЕННЫЙ

Теперь, когда мы увидели, что мы можем сделать с основами с помощью некоторых примеров, давайте рассмотрим некоторые более сложные вещи.

ЭТО ВСЕ О КОМБИНАЦИЯХ

Делать эти различные вещи индивидуальными — мощная способность, но настоящая магия tcpdump исходит из способности сочетать варианты креативными способами, чтобы изолировать именно то, что вы ищете. Есть три способа сделать комбинации, и если вы вообще изучали программирование, они вам будут очень знакомы.

AND
and or &&

OR
or or ||

EXCEPT
not or !

Вот несколько примеров комбинированных команд.

ИЗ СПЕЦИФИЧЕСКОГО IP И НАЗНАЧАЕТСЯ ДЛЯ ОПРЕДЕЛЕННОГО ПОРТА

Давайте найдем весь трафик с 10.5.2.3 к любому хосту на порте 3389.

```
# tcpdump -nnvvS src 10.5.2.3 and dst port 3389
```

ОТ ОДНОЙ СЕТИ К ДРУГОЙ

Давайте посмотрим на весь трафик, идущий от 192.168.x.x, и двигающийся к сетям 10.x или 172.16.x.x, и мы покажем шестнадцатиричный вывод без имени хоста и один уровень дополнительной детализации.

```
# tcpdump -nvX src net 192.168.0.0/16 and dst net 10.0.0.0/8 or172.16.0.0/16
```

НЕ ICMP ТРАФИК, ПЕРЕХОДЯЩИЙ В СПЕЦИФИЧЕСКИЙ IP

Это покажет нам весь трафик, идущий к 192.168.0.2, который не является ICMP.

```
# tcpdump dst 192.168.0.2 and src net and not icmp
```

ТРАФИК ОТ ХОСТА, КОТОРЫЙ НЕ В КОНКРЕТНОМ ПОРТУ

Это покажет нам весь трафик от хоста, который не является трафиком SSH (если предположить использование порта по умолчанию).

```
# tcpdump -vv src mars and not dst port 22
```

Как вы можете видеть, вы можете создавать запросы, чтобы найти практически все, что вам нужно. Ключ должен сначала определить именно то, что вы ищете, а затем построить синтаксис, чтобы изолировать определенный тип трафика.

Сложная группировка и специальные символы

Также имейте в виду, что при создании сложных запросов вам, возможно, придется группировать свои параметры, используя одинарные кавычки. Одиночные кавычки используются для того, чтобы указать tcpdump, что нужно игнорировать некоторые специальные символы – в этом случае то, что в скобках «()». Этот же метод можна использоваться для группировки с использованием других выражений, таких как хост, порт, сеть и т.д. Посмотрите на приведенную ниже команду.

```
# Traffic that’s from 10.0.2.4 AND destined for ports 3389 or
```

22 (неверно)
tcpdump src 10.0.2.4 and (dst port 3389 or 22)

Если вы попытались выполнить эту очень полезную команду в другом случае, вы получите ошибку из-за скобок. Вы можете исправить это, выйдя из скобок (поставив перед каждой из них \) или поставив всю команду в одинарные кавычки:

Traffic that’s from 10.0.2.4 AND destined for ports 3389 or 22 (correct)
tcpdump 'src 10.0.2.4 and (dst port 3389 or 22)'

Изолирование специфических TCP-флагов

Вы также можете захватывать трафик на основе определенных флагов (-ов) TCP.

[ПРИМЕЧАНИЕ: Фильтры ниже находят эти различные пакеты, потому что tcp [13] замечает смещение 13 в заголовке TCP, число представляет местоположение в байте, а ! = 0 означает, что данный флаг установлен в 1, т.е. он включен.]

Показать все URGENT (URG) пакеты …

tcpdump 'tcp[13] & 32!=0'

Показать все ACKNOWLEDGE пакеты (ACK) …

tcpdump 'tcp[13] & 16!=0'

Показать все PUSH пакеты (PSH) …

tcpdump 'tcp[13] & 8!=0'

Показать все RESET пакеты (RST) …

tcpdump 'tcp[13] & 4!=0'

Показать все SYNCHRONIZE пакеты (SYN) …

tcpdump 'tcp[13] & 2!=0'

Показать все FINISH (FIN) пакеты …

```
# tcpdump 'tcp[13] & 1!=0'
```

Показать все SYNCHRONIZE / ACKNOWLEDGE пакеты (SYNACK) …

```
# tcpdump 'tcp[13]=18'
```

[Примечание: только флаг PSH, RST, SYN и FIN отображаются в выводе поля tcpdump. Отображаются URG и ACK, но они показаны в другом месте на выходе, а не в поле flags.]

Однако, как и в случае с самыми мощными инструментами, существует множество способов сделать то, что нужно. В следующем примере показан другой способ захвата пакетов со специфическими наборами TCP-флагов.

```
# tcpdump 'tcp[tcpflags] == tcp-syn'
```

Снять флаги RST с помощью параметра tcpflags …

```
# tcpdump 'tcp[tcpflags] == tcp-rst'
```

Снять флаги FIN с помощью параметра tcpflags…

```
# tcpdump 'tcp[tcpflags] == tcp-fin'
```

[Примечание: тот же метод может быть использован и для других флагов; они были опущены в интересах экономии места.]

Определение заслуживающего внимания трафика

Наконец, есть несколько быстрых рецептов, которые вы захотите запомнить, чтобы поймать специфический и специализированный трафик, например, неправильные / вероятно-вредоносные пакеты.

ПАКЕТЫ С ОБЫЧНЫМИ КОМПЛЕКТАМИ RST И SYN (ЭТОГО НЕ ДОЛЖНО БЫТЬ)

```
# tcpdump 'tcp[13] = 6'
```

НАЙТИ ОТКРЫТЫЙ ТЕКСТ HTTP И ПОЛУЧИТЬ ЗАПРОС

```
# tcpdump 'tcp[32:4] = 0x47455420'
```

НАЙДИТЕ SSH-СОЕДИНЕНИЯ НА ЛЮБОЙ ПОРТ (ЧЕРЕЗ БАННЕР)

```
# tcpdump 'tcp[(tcp[12]>>2):4] = 0x5353482D'
```

ПАКЕТЫ С TTL МЕНЬШЕ 10 (КАК ПРАВИЛО ПОКАЗЫВАЕТ ПРОБЛЕМУ ИЛИ ИСПОЛЬЗУЕТ TRACEROUTE)

```
# tcpdump 'ip[8] < 10'
```

ПАКЕТЫ С УСТАНОВКОЙ EVIL BIT

```
# tcpdump 'ip[6] & 128 != 0'
```

tcpdump — полезное руководство с примерами