IPNAT tune - тюнинг IPNAT

Опубликовано muff в Втр, 2010-05-11 03:07

Есть в наличии маршрутизатор на базе PC с установленной FreeBSD, который рулит несколько подсетей, с общими потоками трафика около двухсот мегабит. В большой и злой интернет пользователи попадают через NAT, реализованый с помощью IPNAT. Все пользуют интернет, все довольны... Недоволен только админ, и поэтому решил заняться оптимизацией работы IPNAT.

Перелопатив немного информации, было обнаружено, что в дефолтной конфигурации IPNAT поддерживает в таблице трансляции только 30 000 записей. Конечно, это немало, но не будем забывать о том, что торренты в сети никто не отменял. И действительно, запросив статус, увидел что перспективы не такие уже и радужные (параметр "inuse" - и это в два часа ночи!):

```
# ipnat -s
mapped  in      2323613704      out     3953865635
added   873951994       expired 285926343
no memory       419891  bad nat 3265285
inuse   19793
orphans 0
rules   24
wilds   0
hash efficiency 10.34%
bucket usage    100.00%
minimal length  2
maximal length  21
average length  9.669
TCP Entries per state
    0     1     2     3     4     5     6     7     8     9    10    11
    0  1037     1     0   280   107    92     0     2     0  3031   202
```

Посмотрим, сколько сесий ориентировочно открывает один пользователь (и познакомимся с самыми злосными качальщиками):

```
# ipnat -l | awk '{print $2}' | sort | uniq -c | sort -r | grep "10.200." | less
 2767 10.200.106.19
2079 10.200.121.15
1484 10.200.126.8
1312 10.200.126.15
 799 10.200.108.5
 685 10.200.108.16
 652 10.200.126.6
 625 10.200.119.3
 598 10.200.114.23
 580 10.200.122.7
 578 10.200.128.4
 519 10.200.108.6
 252 10.200.0.2
 249 10.200.123.10
 222 10.200.121.2
 168 10.200.97.2
 154 10.200.97.12
 128 10.200.121.22
 118 10.200.107.2
 109 10.200.125.9
 108 10.200.121.18
```

```
105 10.200.127.7
 86 10.200.97.18
 49 10.200.124.4
 48 10.200.116.2
 14 10.200.116.16
 12 10.200.119.10
 10 10.200.121.20
  7 10.200.119.2
  3 10.200.119.6
  2 10.200.125.5
  2 10.200.120.2
  2 10.200.106.3
  2 10.200.106.14
  1 10.200.113.4
  1 10.200.108.9
  1 10.200.108.19
```

Как видим, у некоторых "юзверей" количество сессий достигает до трех тысяч! Проверим, сколько все-таки соединений может удерживать IPNAT в текущей конфигурации:

```
# ipf -T list | grep nattable
ipf_nattable_sz min 0x1 max 0x7fffffff current 2047
ipf_nattable_max min 0x1 max 0x7fffffff current 30000
```

Ну вот... Максимальное количество записей в таблице - 30 000. Явно пора заняться "оптимизацией".

Решить эту задачу можно двумя способами. Какой из них выбрать - решать вам. Лично я использую второй.

Способ 1. ***Использование LARGE_NAT.***

Открываем файл /usr/src/sys/contrib/ipfilter/netinet/ip_nat.h и ищем словосочетание #undefine LARGE_NAT (#undef LARGE_NAT) и меняем на #define LARGE_NAT. Включением этой функции увеличивается значение количества сессий до 180 000 (после пересборки ядра). Однако не обязательно останавливаться только на этом, советую изменить еще несколько параметров:

```
// # define NAT_SIZE 2047
# define NAT_SIZE 4093
// # define RDR_SIZE 2047
# define RDR_SIZE 4093

// # define HOSTMAP_SIZE 8191
# define HOSTMAP_SIZE 16383

// # define NAT_TABLE_MAX 180000
# define NAT_TABLE_MAX 300000

// # define NAT_TABLE_SZ 16383
# define NAT_TABLE_SZ 32765
```

Первая строка - старые значения, вторая - новые (увеличенные почти в два раза). После внесенныя изменений пересобираем ядро и наслаждаемся новыми переменными.

Однако обнаружена досадная закономерность - рвутся сессии НАТа, если они неактивны 10 минут. Как выяснилось, дело именно в определении LARGE_NAT. Если посмотреть внимательно код в файле ip_nat.c, то можно увидеть, что время обрыва сессии принудительно устанавливается в 10 минут, если включён LARGE_NAT. Итак, порыскав немного по Сети, нашел информацию о том, что LARGE_NAT используеся, когда мы имеем тысячи

подсетей. Особый смысл в этом параметре - именно ограничение сессии в 10 минут, посколько для тысяч сетей колличество одновременно открытых NAT-сессий может достигать миллионов. Соотвественно, таблица NAT'а быстро бы переполнялась. Поэтому немного подумав, вернул обратно значение #undefine LARGE_NAT.

Ну а увеличение значения переменных дает только позитивный результат, на чем мы и остановимся.

Способ 2. *Использование IPFilter.*

Итак, для того чтобы IPNAT работал, достаточно только двух строк в rc.conf:

```
ipnat_enable="YES"
ipnat_rules="/etc/ipnat.rules"
```

Теперь "подключим" дополнительно ipfilter [1] и немножко "оттюнингуем" его. Для этого внесем следующие изменения в rc.conf:

```
ipfilter_enable="YES"
ipfilter_rules="/etc/ipf.rules"
ipfilter_flags="-D -T ipf_nattable_sz=10009,ipf_nattable_max=300000 -E"
ipnat_enable="YES"
ipnat_rules="/etc/ipnat.rules"
```

По умолчанию ipfilter запускается с правилом по умолчанию block all (пропускать все). Ну а поскольку мы будем использовать только для того, чтобы IPNAT мог устанавливать и удалять свои правила NAT, то этого достаточно. Создадим пустой файл конфигурации:

```
# touch /etc/ipf.rules
```

После этого перезапускаем ipfilter и ipnat:

```
# sh /etc/rc.d/ipfilter start
Enabling ipfilter.
# sh /etc/rc.d/ipnat restart
0 entries flushed from NAT table
0 entries flushed from NAT list
Installing NAT rules.
0 entries flushed from NAT table
0 entries flushed from NAT list
```

Проверям максимальное количество записей в таблицу трансляции:

```
# ipf -T list | grep nattable
ipf_nattable_sz min 0x1 max 0x7fffffff current 10009
ipf_nattable_max min 0x1 max 0x7fffffff current 300000
```

Проверим работу IPNAT:

```
# ipnat -s
mapped in 66440135 out 57477607
added 1065461 expired 990848
no memory 0 bad nat 15644
inuse 41106
orphans 0
rules 23
wilds 0
hash efficiency 23.99%
bucket usage 98.52%
minimal length 0
maximal length 13
```

```
average length 4.169
TCP Entries per state
0 1 2 3 4 5 6 7 8 9 10 11
3 4181 14 0 1619 3056 956 0 12 0 4313 441
```

Ну вот, таблица трансляции уже перевалила за предыдущий лимит в 30 000 сесий, причем изрядно. На этом, кажется, все... Разве что можно задуматься еще над "ужесточением" трансляции за счет понижения таймаутов соединения. Пример флагов IPFilter в таком случае:

```
ipfilter_flags="-D -T ipf_nattable_sz=10009,ipf_nattable_max=300000,\
fr_tcptimeout=180,fr_tcpclosewait=60,fr_tcphalfclosed=7200,fr_tcpidletimeout=172800 -E"
```

Источник (получено *2025-11-12 01:30*): http://muff.kiev.ua/content/ipnat-tune-tyuning-ipnat

Ссылки:
[1] http://www.freebsd.org/doc/ru/books/handbook/firewalls-ipf.html