Обзор и сравнение способов настройки NAT на FreeBSD

В этой статье я бы хотел привести примеры настройки NAT на ОС FreeBSD и провести некоторое сравнение способов, которые, по моему мнению, наиболее часто используются.

Для начала:
*NAT (от англ. Network Address Translation — «преобразование сетевых адресов») — это механизм в сетях TCP/IP, позволяющий преобразовывать IP-адреса транзитных пакетов. Также имеет названия IP Masquerading, Network Masquerading и Native Address Translation.*Рассмотренные варианты:
— Демон Natd
— IPFilter (ipnat)
— PF nat
— ng_nat
— ipfw nat (kernel nat)

NAT с помощью natd

Из хендбука:
Демон преобразования сетевых адресов (Network Address Translation) во FreeBSD, широко известный как natd(8), является демоном, который принимает входящие IP-пакеты, изменяет адрес отправителя на адрес локальной машины и повторно отправляет эти пакеты в потоке исходящих пакетов. natd делает это, меняя IP-адрес отправителя и порт таким образом, что когда данные принимаются обратно, он может определить расположение источника начальных данных и переслать их машине, которая запрашивала данные изначально.

Для работы natd нужен *ipfw*.
В ядре:

```
#Поддержка ipfw
options IPFIREWALL
options IPFIREWALL_VERBOSE
```

```
 options «IPFIREWALL_VERBOSE_LIMIT=100»
 #DIVERT пакетов приходящих на интерфейс для NAT
 options IPDIVERT
```

В /etc/rc.conf добавить
gateway_enable=«yes»

или в */etc/sysctl.conf* добавить
net.inet.ip.forwarding=1.

em0 – внешний интерфейс
192.168.0.0/24 – внутренняя сеть
200.200.200.200 – внешний адрес

Также в /etc/rc.conf добавить:
natd_enable=«YES»
natd_interface=«em0»
natd_flags=””

В фаервол добавляем правила для divert:
/sbin/ipfw add divert natd ip from 192.168.0.0/24 to any out via em0
/sbin/ipfw add divert natd ip from any to 200.200.200.200 in via em0

Более подробно описано в Хендбуке.

NAT с помощью IPFilter (ipnat)

В ядре:

```
 options IPFILTER
 options IPFILTER_LOG
```

или подгрузить как модуль и не трогать ядро.

В */etc/rc.conf* добавить
gateway_enable=«yes»

или в /etc/sysctl.conf добавить
net.inet.ip.forwarding=1.

Также в /etc/rc.conf добавить:
ipnat_enable=«YES» #Включаем ipnat
ipnat_program=”/sbin/ipnat” #Путь к ipnat
ipnat_rules=”/etc/ipnat.rules” #Правила
ipnat_flags=”” #С какими параметрами стартовать

Для ведения логов в *syslog.conf* добавить:
local0.* /var/log/ipmon.log

и запустить утилиту мониторинга работы IPFilter – ipmon с ключами -Dvas
-D – запуститься демоном
-v – детализировать
-a – отслеживать все устройства IPFilter
-s – через syslog

Примеры:

Если:
em0 – внешний интерфейс
192.168.0.0/24 – внутренняя сеть
200.200.200.200 – внешний адрес

То пример правил для Нат будет выглядеть так:
map em0 from 192.168.0.0/24 to any -> 200.200.200.200/32

Или без конкретизации адресов назначения:
map em0 192.168.0.0/24 -> 200.200.200.200/32

Если адрес динамический то можно сделать так:
map em0 192.168.0.0/24 -> 0.0.0.0/32

Некоторые полезные команды при работе с ipnat:
Перезагрузка ipnat:
/etc/rc.d/ipnat restart
Общая статистика роботы Нат:
ipnat –s
Список активных правил и список активных в данный момент сеансов:

ipnat –l
Перечитать конфиг:
ipnat -CF -f /etc/ipnat.rules
-C – очищает таблицу правил.
-F – удаляет записи из таблицы трансляций.
Больше об ipnat и IPFilter в целом можно узнать из:
ipnat(1), ipnat(5), ipnat(8), ipf(5), ipf(8), ipfstat(8), ipftest(1), ipmon(8)
Более подробно здесь.

NAT с помощью pf

В ядре:

```
device pf # Включаем PF OpenBSD packet-filter firewall
device pflog # Поддержка логов pf
```

В */etc/rc.conf* добавить
gateway_enable=«yes»

или в */etc/sysctl.conf* добавить
net.inet.ip.forwarding=1.

Также в */etc/rc.conf* добавить:
pf_enable=«YES»
pf_rules=”/etc/pf.conf”
pf_program=”/sbin/pfctl”
pf_flags=””
pflog_enable=«YES»
pflog_logfile=”/var/log/pf.log”
pflog_program=”/sbin/pflogd”
pflog_flags=””

Пример самого правила:
em0 – внешний интерфейс
192.168.0.0/24 – внутренняя сеть
200.200.200.200 – внешний адрес

В /etc/pf.conf:
nat on em0 from 192.168.0.0/24 to any -> (em0)

NAT с помощью ng_nat

В ядре:

```
 options NETGRAPH
 options NETGRAPH_IPFW
 options LIBALIAS
 options NETGRAPH_NAT
 …и другие опции нетграфа если надо
```

Или просто подгрузить модули:
/sbin/kldload /boot/kernel/ng_ipfw.ko
/sbin/kldload /boot/kernel/ng_nat.ko

em0 – внешний интерфейс
192.168.0.0/24 – внутренняя сеть
200.200.200.200 – внешний адрес

Создание NAT ноды:
ngctl mkpeer ipfw: nat 60 out
ngctl name ipfw:60 nat
ngctl connect ipfw: nat: 61 in
ngctl msg nat: setaliasaddr 200.200.200.200
В ipfw добавляем строчки для перенаправления трафика в созданную ноду:
/sbin/ipfw add netgraph 61 all from any to 200.200.200.200 in via em0
/sbin/ipfw add netgraph 60 all from 192.168.0.0/24 to any out via em0
далее
sysctl net.inet.ip.fw.one_pass=0

Все написанное оформить виде скрипта и засунуть в /usr/local/etc/rc.d с правами запуска для автозагрузки.

Более подробно здесь.

NAT с помощью ipfw nat

Поддержка ipfw nat появилась начиная с версии FreeBSD 7.0

В ядро:

```
 options IPFIREWALL
 options IPFIREWALL_DEFAULT_TO_ACCEPT
 options IPFIREWALL_FORWARD
 options IPFIREWALL_VERBOSE
 options IPFIREWALL_VERBOSE_LIMIT=50
 options IPFIREWALL_NAT
 options LIBALIAS
```

В /etc/rc.conf добавляем

```
 firewall_enable=«YES»
 firewall_nat_enable=«YES»
 firewall_type="/etc/firewall"
 gateway_enable=«YES»
```

В /etc/sysctl.conf добавить:

```
net.inet.ip.fw.one_pass=1
```

em0 – внешний интерфейс
192.168.0.0/24 – внутренняя сеть
200.200.200.200 – внешний адрес

Пример:

```
/sbin/ipfw add nat 1 config log if em0 reset same_ports
/sbin/ipfw add nat 1 ip from 192.168.0.0/24 to not table\(10\)
via em0
/sbin/ipfw add nat 1 ip from any to 200.200.200.200 via em0
```

Где table 10 – не идет через нат

Некоторую статистику можно посмотреть так:
ipfw nat 1 show

Немного сравнения

Следует сказать, что ipfw, natd, ipf, ipnat отлично уживаются вместе. При этом нужно помнить особенности фильтров: ipfw срабатывает по первому совпадению, а ipf (без опции quick в

правиле) – по последнему. Ну и всегда следует иметь в виду порядок прохождения пакета через фильтры. Так, если поддержка ipf собрана в ядре, то независимо от того, как запущен ipfw, в первую очередь пакеты будут проходить через правила ipf, а ipfw получит на вход только то, что будет им пропущено. Если же ipfw собран в ядре, а ipf подгружен как модуль, то правом первенства будет пользоваться ipfw.

Если рассматривать разницу и особенности то можно отметить следующее:

Natd:
– ~~Подыхает~~ становится не эффективен, когда трафик превышает 40-50 мегабит
– Реализация в виде демона
– Сложности при работе на нескольких интерфейсах
+ Простота настройки
+ Функциональность, гибкость

IPnat:
+ Простота настройки
+ «близость» к ядру
– При очень больших нагрузках нужен тюнинг

ng_nat:
– Сравнительно сложная настройка
– Не умеет redirect_port
+ Реализирован через libalias в ядре
+ Потребляет сравнительно немного ресурсов

Ipfw nat:
+ Скорость работы
+ Гибкость
+ Реализирован через libalias в ядре
UPD от nightfly:
– невозможность нормально без тонны алиасов натить из под пула
– невнятная статистика
+ наконец перестало течь

+ в отличии от иных умеет активное фтп изкоробки что позволяет не держать рядом фтп прокси
+ не страдает детскими болезнями типа приколов с одновременными pptp сквозь nat

Pf nat:
+ Скорость работы
+ Использование макросов для написания правил
– Проблемы с smp

Выводы

Я не буду отмечать никакой из выше описанных вариантов ввиду того, что тема немного холиварная, и мнения читателей могут не совпадать с моим. Я только попытался описать примеры настройки и сделать некоторое сравнение множества методов организации NAT на ОС FreeBSD. Также я не стал описывать матчасть по NAT, потому, что она есть здесь.