Тема № 3. NAT – NETWORK ADDRESS TRANSLATON

- RFC 1631 – The IP Network Address Translator (NAT)
- RFC 2663 – IP Network Address Translator (NAT) Terminology and Considerations
- RFC 3022 – Network Address Translation (Traditional NAT)

Технологията NAT е предложена (май от Cisco) за решаване на проблема с изчерпването на **уникалните** IP адреси, предназначени за раздаване и връзка между възлите в Internet.

Възможни решения на този проблем са:

1. проектиране на нова версия на IP протокола (IPv6) – изисква дълго време за разработване и внедряване;
2. преминаване към безкласова адресация и маршрутизиране (CIDR), позволяващо по-рационално използване на съществуващите IP адреси – по времето, когато се взима решението за такъв преход, по-голямата част от мрежите от класове A и B вече са раздадени;
3. използване на публични и частни IP адреси и тяхното транслиране.

Използването на NAT позволява решаването на следните проблеми:

1. осигуряване на корпоративните и частни локални мрежи с голям брой частни IP адреси, като при това не възникват конфликти от използването на еднакви мрежови адреси от различни организации;
2. осигуряване на допълнителна безопасност на възлите от вътрешната мрежа чрез тяхното скриване от външната мрежа, един вид защитна стена[1];
3. организиране на достъпа до Internet през един-единствен шлюз с използване на един единствен **уникален** IP адрес.

Шлюз (aka gateway) – крайно устройство, което е едновременно свързано към повече от една мрежа. Всеки шлюз трябва да има присвоен IP адрес от всяка мрежа, към която е свързан.

Защо са необходими шлюзовете? Защото гледната точка на възел от Ethernet мрежа е ограничена – той може да обменя информация само с възлите от мрежата, в която се намира. Достъпът до всички останали възли се извършва само през машини със специално предназначение – шлюзовете.

Освен гореизброените проблеми, съвременните реализации на технологията за NAT позволяват решаването и на различни други задачи, засягащи администрацията на мрежите:

1. контрол и отчитане на трафика на крайните потребители към Internet;
2. наблюдение на поведението им в мрежата;
3. водене на статистика.

Апаратна реализация на NAT е вградена в различни DLS, ADSL и кабелни модеми, маршрутизатори и комутатори.

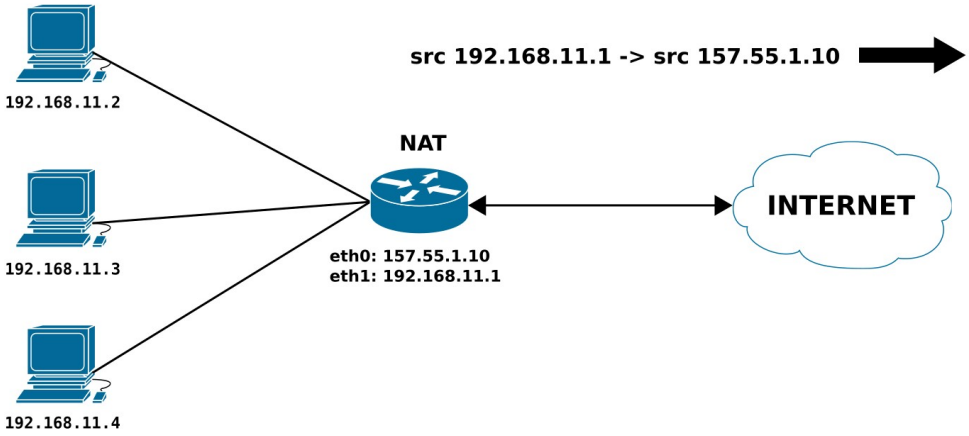
Технологията NAT описва процеса на модифициране на мрежовия адрес (а понякога и на портовете), съдържащи се в заглавните части (header) на IP пакетите при тяхното предаване по мрежата.

---

1 Макар, че идеята за security by obscurity не е най-доброто нещо. Но дали имаме чисто obscurity :)

Технологията NAT позволява на едно единично устройство, обикновено маршрутизатор, да играе ролята на агент между локалната (частна) мрежа и Internet, което означава, че е необходим само един, уникален IP адрес за представяне на цялата група възли от локалната мрежа.

Нека имаме следната конфигурация:



Фигура 1. Принцип на действие на NAT

При изпращането на IP пакет от 192.168.11.3 към Internet, в неговата заглавна част (header) е записан адреса на източника (source address) – 192.68.11.3. В NAT сървъра се извършва транслацията на този адрес от 192.168.11.3 на 157.55.1.10. Обикновено възелът, който извършва това преобразуване изпълнява и функцията на защитна стена и граничен маршрутизатор.

При получаване на отговор на изпратения IP пакет неговият адрес на получател няма да бъде 192.168.11.3, а ще бъде 157.55.1.10. Решението е използване на вградена в NAT сървъра таблица, пазеща информация за всяка създадена връзка с използване на номерата на порт на източника. Всеки TCP или UDP пакет съдържа в себе си две 16 битови числа, указващи номерата на портове на източника и на получателя.

Мрежови порт – програмна абстракция, използвана за определяне на различните крайни точки на комуникационните канали в рамките на един възел от мрежата.

Мрежовият адрес, заедно с номера на порта идентифицират еднозначно крайната точка на един (комуникационен) канал в рамките на една мрежа. Комбинацията от мрежов адрес и номер на порт се нарича **транспортен адрес** (по OSI модела). Сборът от мрежовите адреси и номерата на портове (по един на възел) на два комуникиращи помежду си възела, идентифицира еднозначно комуникационния канал между тях и се нарича **мрежов цокъл (network socket)[2].**

Портовете се ползват от **транспортния слой** на OSI модела (слой 4).

Два комуникиращи си възела от мрежата могат да имат повече от един комуникационен канал помежду си, като в този случай каналите се различават поне

2 The addresses specify the two machines at each end, while the port numbers ensure that the connection between the two computers has a unique identifier. The combination of these four numbers defines a single TCP/IP connection. Each port number uses 16 bits, which means that there are a possible 65,536 (216) values. Realistically, since different manufacturers map the ports in slightly different ways, you can expect to have about 4,000 ports available.

по един от портовете си. Мрежовият порт може да се разглежда като аналогия на апаратния порт → място на контакт между комуникиращи си процеси от различни възли в мрежата.

Таблицата, поддържана от машината, която извършва NAT може да има 65535 реда (2^{16}), в която се записва двойката source_id ↔ source_port за всеки изходящ пакет. След това се извършва както замяна на IP адреса, така и на оригиналния номер на порт на източника на пакета и той се изпраща по предназначение. Когато се получи отговор, номерът на порт на получателя се използва като индекс в таблицата за NAT за да се извлече IP адресът и номерът на порт на възела от вътрешната мрежа, който трябва да получи този отговор.

Апаратна реализация на NAT е вградена в различни ADSL модеми, маршрутизатори и комутатори. Основният принцип на работа на NAT е следния: в модула, извършващ транслацията на мрежовите адреси е вградена таблица, в която се създава запис за всяка осъществена връзка. В него се съдържа IP адресите и номера на портове на източника и на получателя на изпратения пакет. С помощта на този запис се извърша транслирането на мрежовите адреси. Нека разгледаме следния пример: имаме локална мрежа от компютри с един общ изход към Internet. Един от компютрите от тази мрежа (192.168.11.3 например) осъществява връзка към друг компютър (например www.uni-sofia.bg), намиращ се някъде в глобалната мрежа Internet. Връзката към Internet преминава през маршутизатор. Адресът на компютъра, изпращащ пакета е 192.168.11.3, а адресът на получателя е 62.44.96.22. За всяка осъществена връзка, NAT отваря нов порт. В таблицата за NAT се добавя запис за изпратения пакет – IP address & port number. След това NAT променя полето в заглавната част на пакета (полето Source Address), като премахва адреса на компютъра от локалната мрежа (192.168.11.3), изпращаш пакета и го заменя с неговия публичен адрес (157.55.1.10), през който се осъществява връзката към Internet. Обратният пакет съдържа адрес на получателя 157.55.1.10. В таблицата на NAT се намира коя двойка IP_address и port_number отговаря на получения пакет и полето за destination_address се променя с IP адреса от вътрешната мрежа на възела, изпратил първоначално пакета (http://www.k-max.name/linux/netfilter-iptables-v-linux/).

Недостатъци на NAT

1. Транслирането на мрежови адреси нарушава архитектурния модел на протокола IP, според който всеки IP адрес е уникален, т.е идентифицира един единствен възел в Internet. Цялата софтуерна инфраструктура на Internet се крепи на този факт. С помощта на NAT, хиляди машини може да използват примерно адрес 192.168.11.3;
2. NAT променя характера на Internet – от connectionless ориентирана към connection ориентирана, тъй като NAT поддържа информация за съответствието на всяка връзка, преминала през нея. Един срив на машината с NAT и всички TCP връзки ще бъдат разрушени;
3. NAT нарушава фундаменталното правило за разделяне на отделните слоеве. Излизането на една нова версия на протокола TCP, използваща 32 битови номера на портове или преминаване към друг транспортен протокол ще означава край на използването на тази технология;
4. Някои приложения включват в тялото на текста самите IP адреси. NAT не знае какво има в пакета и следователно не може да смени тези адреси, което ще

доведе до пропадане на връзката с отдалечения възел. Пример за такъв протокол е FTP;

5. Тъй като броя на номерата на портове е 16 битово число, то зад един IP адрес могат да бъдат „скрити“ не повече от 65535 машини;
6. Номерата на портовете са „адреси“ на процеси, а не на възли в мрежата;
7. Маршрутизаторите са длъжни да обработват пакети само от слой 3 на модела OSI;
8. Възлите са длъжни да си взаимодействат директно, без намесата на други възли, променящи IP адреси и номера на портове.

Решения на всички горни недостатъци → преминаване към използване на протокол IPv6.

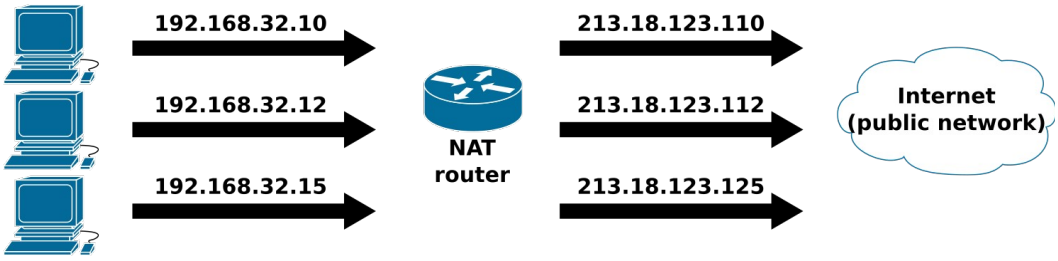
Фиг. 2. Схема на действие на NAT

ВИДОВЕ NAT

Статичен NAT

При статичния NAT имаме конфигурирани определен брой публични IP адреси, отговарящи на определен брой IP адреси от вътрешната мрежа.

Присъединяването (mapping) на локалните адреси към публичните е едно към едно – един локален адрес се заменя винаги с един и същ публичен адрес. Този подход е полезен, когато има нужда от достъп отвън до някой възел вътре в мрежата..

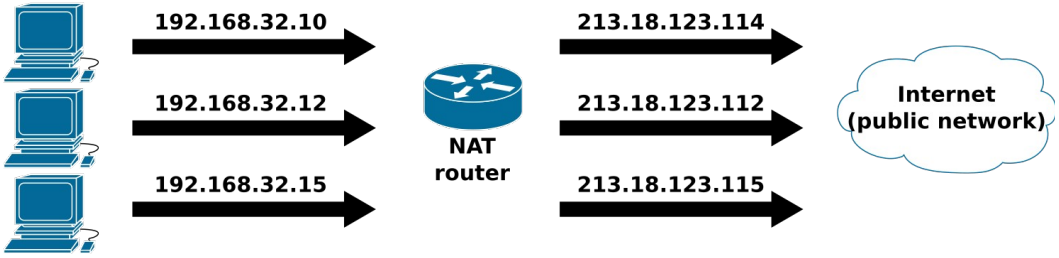


Фиг. 3 Схема на действие на статичния NAT

При статичния NAT, компютърът с адрес 192.168.32.10 винаги ще бъде транслиран до 213.18.123.110. По този начин имаме ясно изразена връзка между локалната и публичната мрежа (Internet).

Динамичен NAT

213.18.123.112	192.168.32.12
213.18.123.113	192.168.32.31
213.18.123.114	192.168.32.7
213.18.123.115	192.168.32.11
213.18.123.116	192.168.32.10

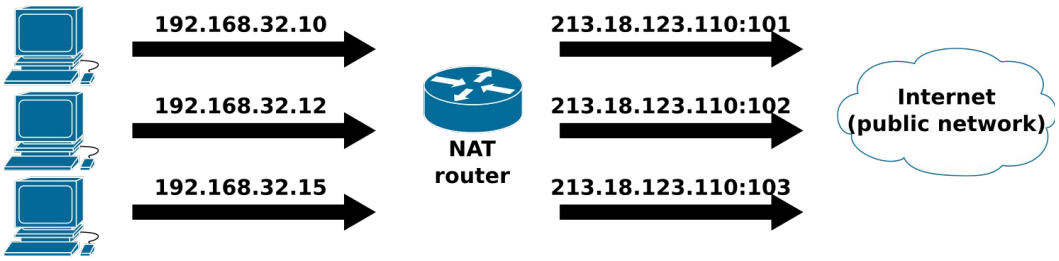


Фиг. 4 Схема на действие на динамичния NAT

При динамичния NAT, адресът на възела от вътрешната мрежа ще бъде транслиран към първия свободен адрес от отредения за транслиране обхват от публични адреси.

ПРЕТОВАРВАНЕ (OVERLOADDING)

Това е разновидност на динамичния NAT.



Фиг. 5. Схема на действие на overloading

При него, множеството адреси от локалната мрежа се транслират към един-единствен публичен адрес, но с различни номера на портове.

Претоварването се използва от свойството на TCP/IP стека, наречено мултиплексиране (multiplexing), което позволява на компютъра да поддържа няколко конкурентни връзки с отдалечения компютър (или компютри), използвайки различни TCP или UDP портове.

Всеки IP пакет има заглавна част, (header), която съдържа следната информация:

- **Адрес на източник (Source Address)** - IP адресът на изпращащия пакета компютър, примерно 201.3.83.132;
- **Номер на порт на източника (Source Port)** – номерът на TCP или UDP порт, присъединен от изпращащия компютър на този пакет, примерно порт 1080;
- **Адрес на получател (Destination Address)** - IP адресът на получаващия пакета компютър, примерно 145.51.18.223;
- **Номер на порт на получателя (Destination Port)** – Номерът на TCP или UDP

порт, който изпращащия компютър е поискал да бъде отворен от получаващия компютър, примерно порт 3021.

Netfilter – пакетния филтър в Centos, с помощта на който се извършва транслацията на мрежови адреси (NAT). Той представлява рамка (framework), даваща ни достъп до пакетите извън стандартния сокет интерфейс на unix/linux.

Iptables – потребителски инструмент за дефиниране на правила за филтриране на пакети и транслиране на мрежови адреси.

На практика, често под името iptables се разбира цялата инфраструктура, включваща netfilter, следенето на връзките, транслирането на мрежовите адреси и самия инструмент iptables.

Дефинираните правила се групират във вериги, които от своя страна се групират в таблици.

ПРАВИЛА→ ВЕРИГИ → ТАБЛИЦИ

Всяко правило дефинира условие и цел. Целта се прилага върху всички пакети, които удовлетворяват условието.

Всеки пакет преминава поне през една верига и се сравнява последователно с правилата от веригата. Ако се получи съвпадение, обхождането на веригата се прекратява и се прилага съответното правило.

Ако пакетът не удовлетвори нито едно условие, тогава върху него се прилага политиката по подразбиране на веригата.

Цел на някое право може да бъде нова верига. Ако пакетът премине през втората верига без да удовлетвори нито едно условие от нейните правила, се продължава с първата верига. **Всяка верига представлява подреден списък с правила.**

Веригите се групират в таблици – всяка таблица е свързана с различен вид обработка на пакетите. Няма ограничения за влагането на веригите една в друга.

В netfilter има три основни вериги – input, output и forward.

Могат да бъдат създавани нови вериги.

Три основни таблици – filter, nat и mangle.

Таблица filter – използва се за филтриране на пакетите (блокиране (drop) или разрешаване (accept)).

Тази таблица има три вериги:

INPUT – всички пакети, отиващи към защитната стена преминават през тази верига;

OUTPUT – всички пакети, излизащи от защитната стена преминават през тази верига;

FORWARD – през тази верига преминвата всички пакети, които преминава през защитната стена (такива пакети, които се маршрутизират).

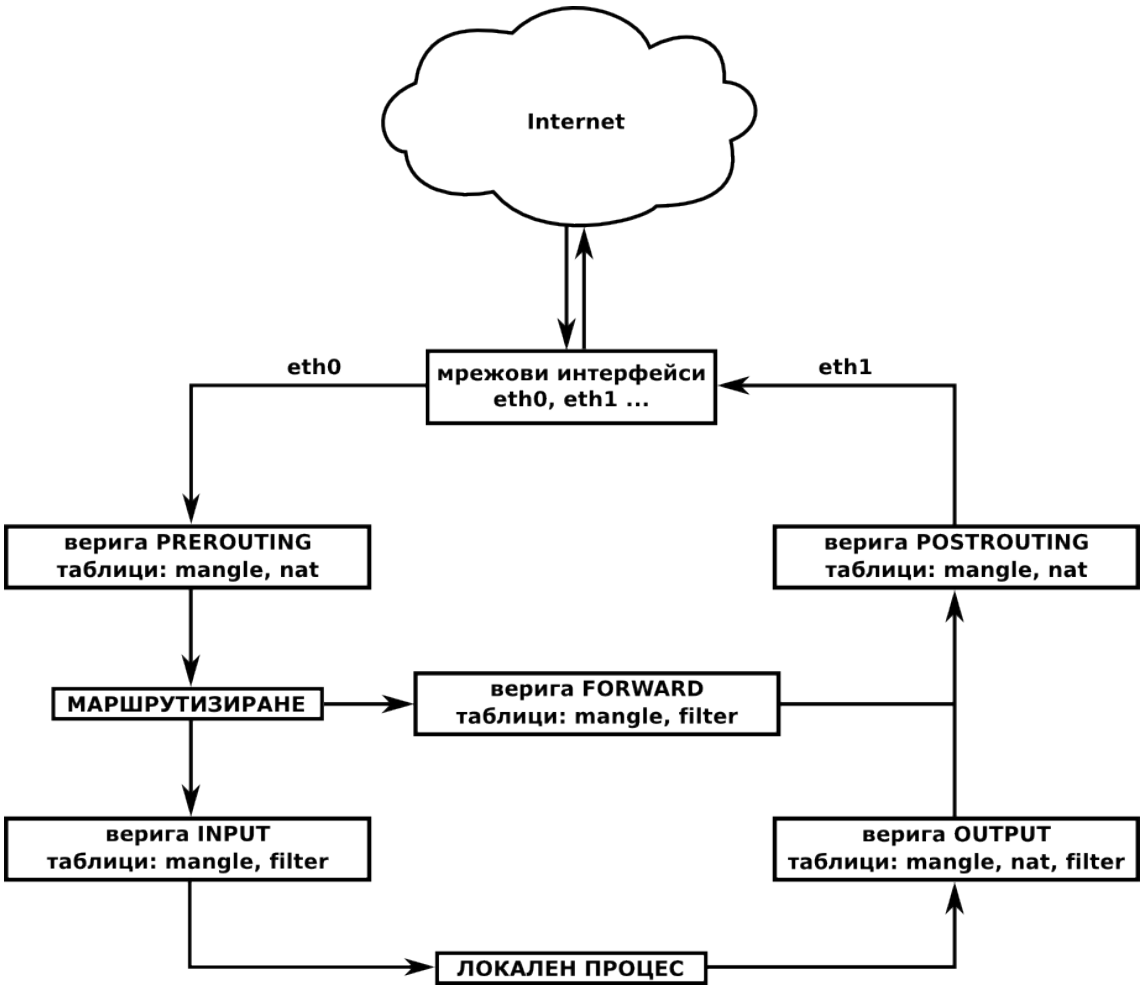
Таблица nat – съхранява правилата, използвани за промяна на мрежовите адреси и номерата на портове (NAT). През тази таблица винаги преминава първият пакет на всяка нова връзка. Съдържа също три вериги:

PREROUTING – входящите пакети преминават през нея преди да се вземе решение за тяхното маршрутизиране. В тази верига се осъществява така наречения DNAT (Destination Network Address Translation) – промяна на адреса на получателя на

пакета;

POSTROUTING – през нея преминават изходящите пакети, след като е взето решение за тяхното маршрутизиране. Тук се осъществява така наречения SNAT (Source Network Address Translation) – промяната на source адреса на изпращача на пакета;

OUTPUT – в тази верига се извършва ограничен DNAT над локално генерираните пакети.



Фиг. 6. Схема на действието на пакетния филтър netfilter

Таблица mangle – използва се за промяна на допълнителните полета в заглавната част (ip header) на пакетите. През нея преминават всички пакети. Поради специфичните си задачи съдържа в себе си всички предефинирани вериги:

PREROUTING – през нея преминават всички пакети, влизащи в системата;

INPUT – през нея преминават всички пакети, предназначени за системата;

FORWARD – през нея преминават всички пакети, минаващи транзитно през системата;

OUTPUT – през нея преминават всички пакети, напускащи (и произлизащи от) системата;

POSTROUTING – през нея преминават всички пакети, напускащи системата.

Както вече споменахме, всяко правило има цел. Тя може да бъде друга верига или някоя от предефинираните цели:

ACCEPT – указва на пакетния филтър да приеме пакета. В зависимост от веригата, това означава изпълнение на различни действия над пакета;

DROP – указва на пакетния филтър да отхвърли пакета и да не го обработва повече. На източника на пакета не се изпраща никакъв отговор и за него блокирането се изразява в изтичане на комуникационния таймаут (и разпадане на връзката).

Допълнителни цели – в ръководството на iptables.

MASQUERADING – ограничена форма на промяна на адреса на източника на пакета (SNAT) за мрежи, чийто IP адрес на външния интерфейс е динамичен (получава се от dhcp сървър). В този случай, вместо да се сменя SNAT правилото при смяна на адреса се използва автоматично адресът на изходящия интерфейс при транслирането.

Синтаксис:

iptables [-t table] -[ADC] chain rule-specification [options]

Трябва да се добавят различни примери за правила на iptables

iptables -t nat -A POSTROUTING -s 172.18.0.0/20 -j SNAT --to-source 62.44.102.7

Примери за използване на iptables:

1. Показване състоянието на iptables

Става с използването на следната команда като root:

```
# iptables -L -n -v
```

изходът от тази команда може да изглежда така:

```
Chain INPUT (policy ACCEPT 0 packets, 0 bytes)
 pkts bytes target     prot opt in     out     source destination
Chain FORWARD (policy ACCEPT 0 packets, 0 bytes)
 pkts bytes target     prot opt in     out     source destination
Chain OUTPUT (policy ACCEPT 0 packets, 0 bytes)
 pkts bytes target     prot opt in     out     source destination
```

Той ни показва, че защитната стена (iptables) не е активна. Следващият примерен изход от горната команда е от активирана защитна стена:

```
# iptables -L -n -v
Chain INPUT (policy ACCEPT 8771K packets, 902M bytes)
 pkts bytes target     prot opt in     out     source               destination
70568 5735K DROP        all  --  eth1   *       0.0.0.0/0            0.0.0.0/0
MAC 00:0A:E6:AF:1C:50
 115K   10M ACCEPT     all  --  lo     *       0.0.0.0/0            0.0.0.0/0
 2598  209K DROP        udp  --  *      *       0.0.0.0/0            0.0.0.0/0
! ctstate RELATED,ESTABLISHED udp dpt:123


Chain FORWARD (policy ACCEPT 778M packets, 587G bytes)
 pkts bytes target     prot opt in     out     source               destination
```

```
69184 5928K ACCEPT     tcp  --  *      *       0.0.0.0/0            62.44.102.7
tcp dpt:22
2824K    679M  ACCEPT          all    --    *            *             0.0.0.0/0
62.44.102.206
    0     0 REJECT     tcp  --  *      *       62.44.102.23          0.0.0.0/0
tcp dpt:445 reject-with tcp-reset
    3   144 REJECT     tcp  --  *      *       62.44.102.41          0.0.0.0/0
tcp dpt:445 reject-with tcp-reset
  490 34938 DROP       udp  --  *      *       0.0.0.0/0            62.44.108.39
udp dpt:123
  750 43829 DROP       udp  --  *      *       0.0.0.0/0            62.44.108.38
udp dpt:123
  488 35086 DROP       udp  --  *      *       0.0.0.0/0            62.44.108.37
udp dpt:123
  489 33663 DROP       udp  --  *      *       0.0.0.0/0            62.44.108.36
udp dpt:123
 4528  178K DROP       udp  --  *      *       0.0.0.0/0            62.44.108.35
udp dpt:123
    0     0 DROP       udp  --  *      *       62.44.108.35          0.0.0.0/0
udp spt:123 dpt:123
97403 7803K DROP       udp  --  *      eth1    0.0.0.0/0            0.0.0.0/0
udp dpt:123 ! ctstate RELATED,ESTABLISHED
  73M   54G ACCEPT     all  --  *      *       0.0.0.0/0            62.44.108.46
4579K 1124M ACCEPT     all  --  *      *       0.0.0.0/0            62.44.108.44
4314K 3446M ACCEPT     all  --  *      *       62.44.108.44         0.0.0.0/0
  67M   36G ACCEPT     all  --  *      *       62.44.108.46         0.0.0.0/0
  71M   29G ACCEPT     all  --  *      *       0.0.0.0/0            62.44.108.40
  74M  146G ACCEPT     all  --  *      *       62.44.108.40         0.0.0.0/0
  13M 2063M ACCEPT     all  --  *      *       62.44.108.32/28      0.0.0.0/0
   15M     11G  ACCEPT          all    --    *            *             0.0.0.0/0
62.44.108.32/28
    0     0 ACCEPT     all  --  *      *       62.44.102.1          0.0.0.0/0
57098 3617K ACCEPT     all  --  *      *       0.0.0.0/0            62.44.102.1
 500M   55G ACCEPT     all  --  *      *       62.44.102.2          0.0.0.0/0
 737M  774G ACCEPT     all  --  *      *       0.0.0.0/0            62.44.102.2
 266K   56M ACCEPT     all  --  *      *       62.44.102.4          0.0.0.0/0
 289K  104M ACCEPT     all  --  *      *       0.0.0.0/0            62.44.102.4
   93  8511 ACCEPT     all  --  *      *       62.44.102.5          0.0.0.0/0
47582 3093K ACCEPT     all  --  *      *       0.0.0.0/0            62.44.102.5
  34M   26G ACCEPT     all  --  *      *       62.44.102.6          0.0.0.0/0
  35M   15G ACCEPT     all  --  *      *       0.0.0.0/0            62.44.102.6
  27M   35G ACCEPT     all  --  *      *       0.0.0.0/0            62.44.102.7
    0     0 ACCEPT     all  --  *      *       62.44.102.8          0.0.0.0/0
 101K 5853K ACCEPT     all  --  *      *       0.0.0.0/0            62.44.102.8
3836K  556M ACCEPT     all  --  *      *       62.44.102.9          0.0.0.0/0
```

```
3775K 1329M ACCEPT     all  --  *      *       0.0.0.0/0            62.44.102.9
    0     0 ACCEPT     all  --  *      *       62.44.102.10         0.0.0.0/0
45466 3045K ACCEPT     all  --  *      *       0.0.0.0/0            62.44.102.10
1973K  503M ACCEPT     all  --  *      *       62.44.102.11         0.0.0.0/0
1226K 2034M ACCEPT     all  --  *      *       0.0.0.0/0            62.44.102.11
    0     0 ACCEPT     all  --  *      *       62.44.102.12         0.0.0.0/0
51647 3361K ACCEPT     all  --  *      *       0.0.0.0/0            62.44.102.12
    0     0 ACCEPT     all  --  *      *       62.44.102.13         0.0.0.0/0
39292 2749K ACCEPT     all  --  *      *       0.0.0.0/0            62.44.102.13
    0     0 ACCEPT     all  --  *      *       62.44.102.14         0.0.0.0/0
50230 3272K ACCEPT     all  --  *      *       0.0.0.0/0            62.44.102.14
4453K  768M ACCEPT     all  --  *      *       62.44.102.15         0.0.0.0/0
4422K 6100M ACCEPT     all  --  *      *       0.0.0.0/0            62.44.102.15
    0     0 ACCEPT     all  --  *      *       62.44.102.16         0.0.0.0/0
46230 3102K ACCEPT     all  --  *      *       0.0.0.0/0            62.44.102.16
    0     0 ACCEPT     all  --  *      *       62.44.102.17         0.0.0.0/0
63609 3994K ACCEPT     all  --  *      *       0.0.0.0/0            62.44.102.17
1171K   68M ACCEPT     tcp  --  *      *       0.0.0.0/0            62.44.102.18
tcp dpt:80
    0     0 ACCEPT     all  --  *      *       62.44.102.19         0.0.0.0/0
42867 2914K ACCEPT     all  --  *      *       0.0.0.0/0            62.44.102.19
    0     0 ACCEPT     all  --  *      *       62.44.102.20         0.0.0.0/0
47721 3156K ACCEPT     all  --  *      *       0.0.0.0/0            62.44.102.20
6095K 8167M ACCEPT     all  --  *      *       62.44.102.28         0.0.0.0/0
5730K 5535M ACCEPT     all  --  *      *       0.0.0.0/0            62.44.102.28
7927K 2174M ACCEPT     all  --  *      *       62.44.102.67         0.0.0.0/0
8347K 1343M ACCEPT     all  --  *      *       0.0.0.0/0            62.44.102.67
  15M   27G ACCEPT     all  --  *      *       0.0.0.0/0            62.44.102.79
  19M 5942M ACCEPT     all  --  *      *       62.44.102.79         0.0.0.0/0
  18M 1400M ACCEPT     all  --  *      *       62.44.102.89         0.0.0.0/0
  18M   44G ACCEPT     all  --  *      *       0.0.0.0/0            62.44.102.89
  90M 4252M REJECT     tcp  --  eth1   *       0.0.0.0/0            0.0.0.0/0
tcp multiport dports 25,137,138,139,445 reject-with icmp-host-unreachable
 164K   14M DROP       udp  --  eth1   *       0.0.0.0/0            0.0.0.0/0
udp multiport dports 137,138,139
  44M 2241M REJECT     tcp  --  *      eth1    0.0.0.0/0            0.0.0.0/0
! ctstate RELATED,ESTABLISHED reject-with icmp-host-unreachable
  79M 8159M DROP       udp  --  *      eth1    0.0.0.0/0            0.0.0.0/0
! ctstate RELATED,ESTABLISHED
 184M  286G ACCEPT     all  --  *      eth1    62.44.96.0/19        0.0.0.0/0
  146M      29G  ACCEPT          all    --    eth1      *              0.0.0.0/0
62.44.96.0/19
 614K   31M            icmp --  *      *       0.0.0.0/0            0.0.0.0/0
```

```
icmp type 8 recent: SET name: DEFAULT side: source

Chain OUTPUT (policy ACCEPT 26M packets, 2116M bytes)
 pkts bytes target     prot opt in     out     source               destination
```

където:

- **-L** : показва правилата в iptables;
- **-v** : показва детайлна информация. Тази настройка създава списък с името на мрежовия интерфейс, настройките на конкретните правила и маската за TOS. Показват се и броя на преминалите пакети, както и тяхната големина в байтове. Големината е показана със съответните представки 'K', 'M' или 'G' за 1000, 1,000,000 и 1,000,000,000 множителя съответно.
- **-n** : показва IP адреса и номера на порт в числов формат. Не използва услугата DNS за намиране на съответните на IP адреса имена, което усккорява извеждането на списъка

Ако искаме номерата на правилата в съответната верига да бъдат показани номерирани, тогава използваме следната команда:

iptables -L -n -v –line-numbers

Тя ще ни даде изход, подобен на този:

```
Chain INPUT (policy DROP)
num  target     prot opt source               destination
1    DROP       all  --  0.0.0.0/0            0.0.0.0/0    state INVALID
2    ACCEPT     all  --  0.0.0.0/0            0.0.0.0/0    state
RELATED,ESTABLISHED
3    ACCEPT     all  --  0.0.0.0/0            0.0.0.0/0
4    ACCEPT     all  --  0.0.0.0/0            0.0.0.0/0
Chain FORWARD (policy DROP)
num  target     prot opt source               destination
1    ACCEPT     all  --  0.0.0.0/0            0.0.0.0/0
2    DROP       all  --  0.0.0.0/0            0.0.0.0/0     state INVALID
3    TCPMSS     tcp  --  0.0.0.0/0            0.0.0.0/0     tcp flags:0x06/0x02
TCPMSS clamp to PMTU
4    ACCEPT     all  --  0.0.0.0/0            0.0.0.0/0           state
RELATED,ESTABLISHED
5    wanin      all  --  0.0.0.0/0            0.0.0.0/0
6    wanout     all  --  0.0.0.0/0            0.0.0.0/0
7    ACCEPT     all  --  0.0.0.0/0            0.0.0.0/0
Chain OUTPUT (policy ACCEPT)
num  target     prot opt source               destination
Chain wanin (1 references)
num  target     prot opt source               destination
Chain wanout (1 references)
```

```
num  target     prot opt source               destination
```

Тази настройка е полезна при изтриване или вмъкване на правила в защитната стена.

За да покажем правилата от конкретна верига на iptables използваме следната команда:

```
# iptables -L INPUT -n -v
```

```
# iptables -L OUTPUT -n -v –line-numbers
```

2. Спиране и пускане на iptables:

```
# service iptables stop
# service iptables start
# service iptables restart
```

Може да използваме командата iptables и за изтриване на всички правила:

```
# iptables -F
# iptables -X
# iptables -t nat -F
# iptables -t nat -X
# iptables -t mangle -F
# iptables -t mangle -X
# iptables -P INPUT ACCEPT
# iptables -P OUTPUT ACCEPT
# iptables -P FORWARD ACCEPT
```

където:

- **-F** : изтрива (flushing) всички правила;
- **-X** : изтрива верига.
- **-t table_name** : избира таблица (в случая nat или mangle) и изтрива правилата от нея;
- **-P** : установява политиката по подразбиране на веригата (може да бъде DROP, REJECT или ACCEPT).

3. Изтриване на правилата на iptables

4. Вмъкване на правила на iptables

5. Запазване на въведените правила на iptables

6. Настройване на политиката по подразбиране на iptables

7 ...