Установка мережі малої компанії чи для дому

Установка мережі малої компанії чи для дому

Дмитро Ковальов

Цей документ описує установку і конфіґурування апаратного і програмного забезпечення для побудови мережі комп'ютерів на базі операційної системи Юнікс і Юнікс-подібних систем. Конкретні приклади стосуються системи Лінакс (Дебіан, Мандрейк) і MacOSX.

Щойно закінчивши установку і переконфіґурацію кількох Лінакс систем вдома хочу поділитися своїми "набутками"з області конфіґурування Лінакса, які можуть задовольнити потреби або домашньої мережі, або офісу невеликої компанії.

Ніяких принципово нових положень в цьому документі немає. Вся інформація, що викладена тут, може бути знайдена в надзвичайній кількості примірників і варіацій у мережі, але досі мені не траплялось послідовне викладення всіх положень в одному документі. До того ж не зустрічалось поки-що викладення зазначеного українською мовою.

Зауваження: після кількох змін до конфіґурації домашньої мережі потрібно було також змінити і документацію. Що я і роблю в цьому оновленому документі. Я не видаляю з опису нічого, що стосувалось старішої конфіґурації, а тільки доповнюю опис новими деталями. Тож, на сьогоднішній день (1 квітня, 2002 року, ні це не першоквітневий жарт, а правда :) зміни такі:

- я перейшов від модемного з'єднання з інтернетом до ADSL;
- мій сервер змінився з Дебіана на MacOS X.

Оскільки MacOS X – це дуже подібна до більшості BSD система (насправді центральна Юнікс-складова MacOS X - Darwin - була побудована з FreeBSD), більшість методів конфіґурування Мака можуть бути застосованими також і до інших BSD-похідних.

Кілька областей, які підлягали інтенсивному конфіґуруванню і які описані в цьому документі такі:

- DNS;
- керування дозвоном до провайдера;
- firewall/IP masquerading (NAT в Маку)
- електонна пошта (на базі sendmail).

Зміст

1 Короткий зміст

- Загальний огляд системи

  Короткий опис архітектури системи і апаратних засобів. Деякі діаграми логічної і фізичної структури системи.

- З'єднання з ISP

  Що вам буде потрібно для того, щоб підключитись до провайдера Інтернет-послуг. Конфіґураційні файли для pppd і для connectd - програми керування pppd, яка стартує і зупиняє pppd на вимогу користувачів.

- Налагоджування DNS

  Служба назв доменів (чи DNS) служить для пошуку IP адрес комп'ютерів за їхніми назвами і навпаки. Без неї можна жити. Якщо Ви вкрай не хочете встановлювати DNS-сервер, можете обмежитись конфіґурацією клієнтів і використовувати DNS Вашого провайдера зв'язку з тенетами. Але наявність DNS може значно полегшити життя адміністраторам і до того ж її не так важко встановити, то чому б нам цього не зробити?

- Захист доступу до системи.

  Для постійно підключеної до тенет системи деяка система захисту просто необхідна. Через сервер всі комп'ютери вашої мережі будуть з'єднуватись з всесвітніми тенетами. Так само через незахищений сервер будь-хто з іншого кінця зможе скористуватись відчиненими дверима до ваших файлів і даних. В цьому розділі я розглядаю деякі найпростіші приклади встановлення фільтрації пакетів на вході в локальну мережу з тенет.

- Конфігурація електронної пошти - сервер та клієнт (на базі sendmail)

  Важко собі уявити сучасну комп'ютерну систему без електронної пошти. В той же час програма sendmail зарекомендувала себе як найважча в системній адміністрації область конфіґурування. Але на конкретних прикладах я збираюсь показати, що не такий страшний вовк, як ним лякають. Після однієї-двох конфіґурацій поштових систем, ви зможете робити це без страху для організації будь-якого масштабу.

- Маленькі поради

  Всілякі дрібнички, які не вдалося втиснути в інші розділи.

2 Загальний огляд системи

В моєму випадку загальна система складається із систем які можна віднести до двох типів:

сервер , що забезпечує зв'язок з світовими тенетами (більш відомих широкому загалу як Інтернет) і

сервер чи сервер/клієнт , домашніх директорій користувачів, настільна робоча станція, тощо.

В цьому документі я надалі буду вживати просто "сервер"для комп'ютера першого типу і "клієнт"для комп'ютера другого. Хоча, насправді, будь-який комп'ютерів в мережі може надавати інші послуги (які не розглядаються в даному документі), тобто бути сервером того чи іншого типу для внутрішньої мережі.

Кількість клієнтів не обмежується одним, хоча тут говориться в більшості про клієнта в однині. Це означає всього-навсього, що конфіґурація всіх клієнтів ідентична. Фактично, тип установки описаний в цьому документі, підходить як для невеличкої домашньої мережі, так і дла офісу невеликої компанії/фірми. Кілька років тому (ще в часи Лінакса ядра 1.3.x) я працював у невеликій фірмі (кількість персоналу – біля 30 чоловік). Саме такий тип конфіґурації забезпечував цій фірмі весь зовнішній зв'язок зі світом: електронна пошта, веб-сервіси (вихід назовню – перегляд зовнішніх сторінок тенет персоналом фірми) і власний веб-сервер, який працював на комп'ютері всередині фірми. Весь зв'язок забезпечувався одним модемом 28k, який був підключений виділеною лінією до інтернет-провайдера. Ще один модем використовувався для сервера факсіміле (як для прийому, так і для відправки), і для PPP (*dial-in*) забезпечував персоналу компанії можливість попасти в локальну мережу компанії з дому.

2.1 Архітектура системи

На рисунку 2.1(стор. 3 подана архітектура локальної мережі. З'єднання вгорі (інтерфейс ppp0) – це з'єднання з провайдером послуг тенет. Всі комп'ютери у внутрішній мережі маскуються від зовнішнього світу, тобто IP адреси їх невидимі поза локальною мережею. Це забезпечується програмами, які реалізують трансляцію мережевих адрес (Network Address Translation, див. [natb]): IP Chains в Лінаксі ([Rus]) та natd в MacOS X ([nata]). Про це – в розділі "Захист доступу до системи". Внутрішня мережа побудована з використанням IP адрес так званого "приватного"діапазону (див. [pri]). Власне через застосування трансляції адрес сеґмент адрес може бути будь-яким, але для захисту від помилок краще все-таки вибирати адреси з діапазонів вказаних у RFC-1918 [pri] (Таблиця 2.1 стор. 3).

IP адреси	Маска мережі (netmask)	Клас мережі
10.0.0.0	10.255.255.255	Class A
172.16.0.0	172.31.255.255	Class B
192.168.0.0	192.168.255.255	Class C

Табл. 1: Приватні мережі і IP-адреси

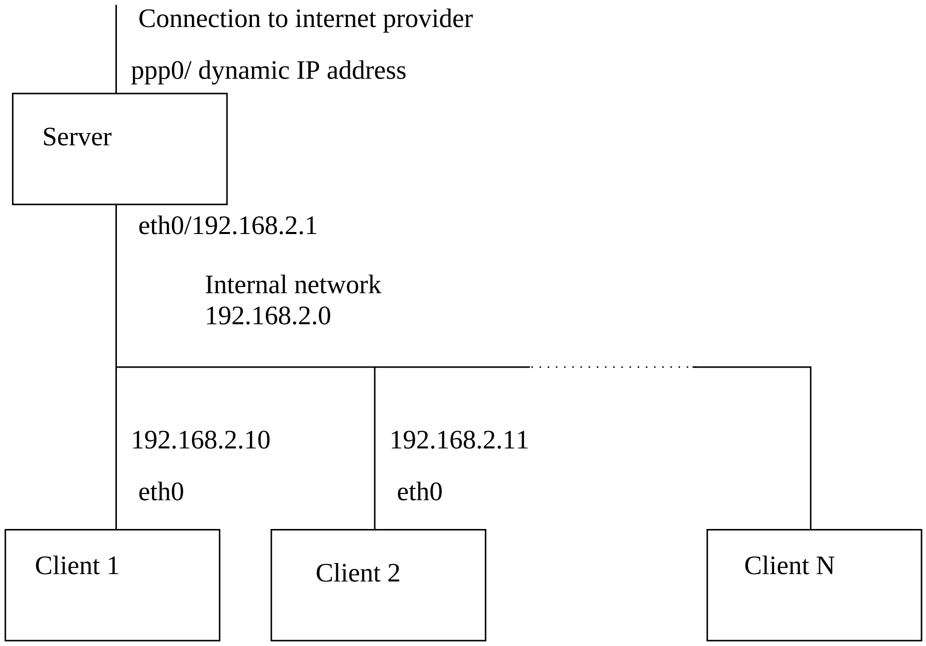

Рис. 1: Загальний вигляд конфіґурації системи.

2.2 Фізична конфіґурація системи

Рисунок 2.1(стор. 3) показує логічну структуру мережі, яка залишається незмінною при всіх конфіґураціях апаратного забезпечення. Апаратна, або фізична конфіґурація мережі залежить від типу з'єднання з інтернет-провайдером (модем, DSL/ADSL, кабельне, опто-волоконне, тощо). Тому далі подані конкретні приклади кількох типів з'єднання – модемного та ADSL.

Зауваження щодо MacOS X: для сервера чи для клієнтів на базі MacOS X не забудьте всі посилання на eth0 (чи будь-який інший ethX) змінити на en0 (чи enX, відповідно). Саме так називається інтерфейс мережі Ethernet в MacOS X.

2.2.1 Модемне з'єднання

При модемному з'єднанні (Рис. 2.2.1) з провайдером архітектура підключень досить прозора – до сервера підключений модем, на якому піднімається інтерфейс ppp0, іншим з'єднанням сервера є інтерфейс внутрішньої мережі ethernet. Всі пакети з ppp0 передаються на eth0 і навпаки. Для того, щоб пакети TCP/IP проходили з одного інтерфейса на інший потрібно вмикнути режим пересилки IP-пакетів [ipf] [1] на сервері.

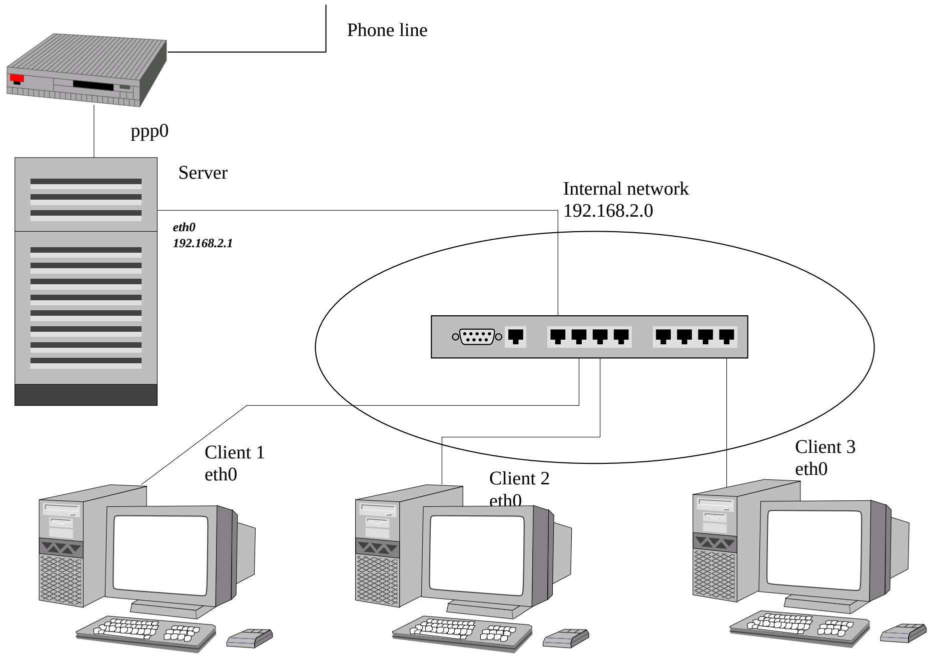

Рис. 2: Конфіґурація мережі. З'єднання з провайдером через модем

2.2.2 Два типи з'єднання ADSL

Якщо використовується ADSL-модем типу маршрутизатора, то з боку комп'ютера ADSL з'єднання виглядає як звичайний 10 мбіт Ethernet. В залежності від кількості мережевих з'єднань сервера, можна розглянути дві відмінні конфіґурації:

[1] IP forwarding англійською

- з двома мережевими платами
- з однією мережевою платою

Для з'єднання з провайдером використовується pppd з драйвером PPPoE [2](див. наприклад Roaring Penguin PPPoE [ppp]).

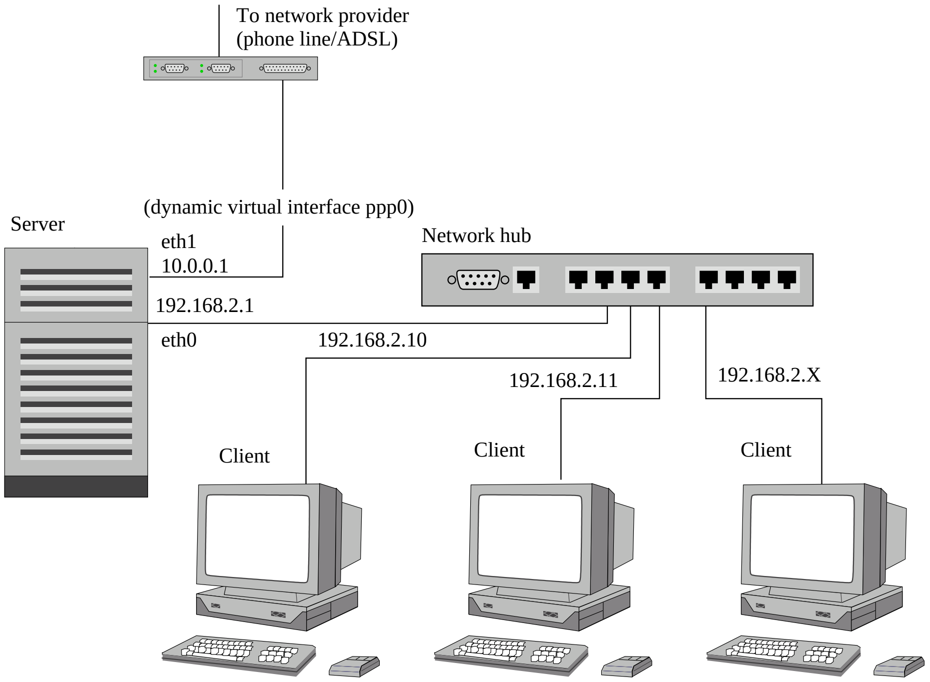

Рис. 3: ADSL з'єднання. Конфіґурація з двома мережевими платами на сервері.

Сервер з двома мережевими платами Цей тип конфіґурації (див. рис. 2.2.2 на стор. 5) дуже подібний до попередньої (модемне з'єднання). Єдиною відчутною різницею є те, що інтерфейс ppp0 піднімається не безпосередньо на послідовному порті, а на вершині ethernet-інтерфейсу. Тобто створюється другий віртуальний інтерфейс (ppp0) на вершині вже існуючого фізичного (eth1). Інтерфейсу eth0 на сервері назначається IP адреса з приватного діапазону 192.168.2.1. eth1 може мати будь-яку іншу IP адресу (бажано з іншого сеґменту, наприклад 10.0.0.1).

Сервер з однією мережевою платою Наступна конфіґурація відрізняється від попередньої тим, що сервер має всього одну мережеву плату. Цією платою сервер підмикається до концентратора, і в цей же концентратор вмикається модем

[2] PPP over Ethernet

ADSL і всі інші комп'ютери домашньої мережі. Хоча фізично така мережа виглядає досить-таки відмінно від попередньої схеми і від схеми з використанням модема, але логічна структура відповідає представленій на рис. 2.2.2. В зовнішній мережі "видно"тільки ту IP-адресу, яку провайдер назначає інтерфейсу ppp0. Всі інші IP-адреси сховані за ppp0. На ppp0 настроюється фільтрація пакетів і вмикається так само режим пересилки IP-пакетів. Пакети пересилаються між двома інтерфейсами ppp0 та eth0 (обидва інтерфейси ppp0 b eth0 є в цьому випадку фізично одним і тим же з'єднанням Ethernet).

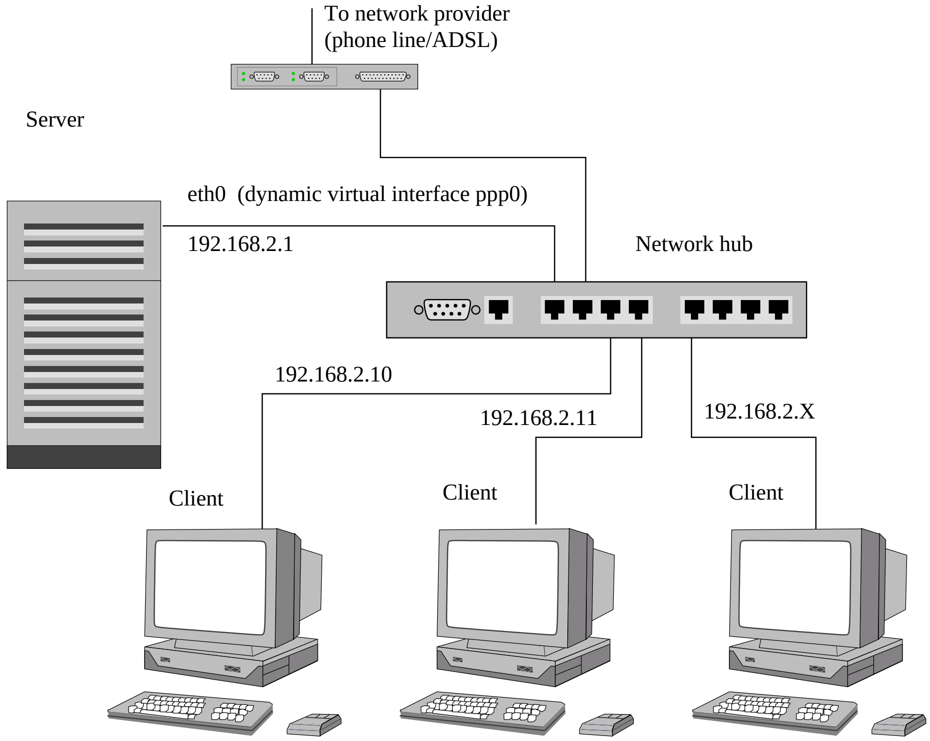

Рис. 4: ADSL з'єднання. Конфіґурація з однією мережевою платою на сервері.

2.3 Операційні системи

При виборі операційної системи для сервера я зупинився на Debian [deb]. Комплект компакт-дисків, які я мав під рукою, були версією 2.2 r3 цієї системи. Тому, саме цю систему я і встановив. Єдине програмне забезпечення, яке мені потрібно було додати до системи, крім того, що малося на компакт-дисках була програма connectd [con]. Звичайно-ж, можна (і, можливо, варто було б) використати замість 2.2 більш нову версію Debian'а [3].

Певний час я користувався сервером на базі MacOS X замість Дебіан'а. MacOSX має в стандартній поставці системи (10.1.2, поновлена до 10.1.5) всі необхідні засоби для встановлення з'єднання PPP (включений стандартний pppd) чи з'єднання ADSL (pppoe), і для настроювання трансляції адрес (в Маку це робить демон natd) і для встановлення захисту-брандмауера ([4] – за допомогою `ipfw`.

Системами-клієнтами в моєму випадку виявились Мандрейк Лінакс (починаючи з версій 6.x і до 8.2 на сьогоднішній день), MacOSX і Debian (i386/PC і Sun Sparc), але при бажанні це може бути будь-яка інша сучасна система, яка задовольняє Вашим вимогам. Оскільки, на початку я користувався більше Мандрейком в якості клієнта, деякі з моїх рекомендацій носять Мандрейк-специфічний характер.

Крім конкретних назв пакетів RPM більшість порад Мандрейка повинні підходити також і до RedHat, але я це особисто не перевіряв. Певна відмінність між укладками Лінакса та Юніксів полягає в rc-скриптах.

Традиційно в Юніксах системи V (SystemV) стартові скрипти розміщаються в каталозі `/etc/init.d`. В каталогах `/etc/rcN.d` (де N – це цифра 1,2,3,4,5,6 чи літера S, що означає робочий рівень системи) створюються символьні чи жорсткі [5] посилання на скрипти виду `S21nfs -> ../init.d/nfs`.

Система Дебіан підтримує саме цю структуру стартових каталогів. RedHat вирішив змінити таку структуру, ввівши ще один підрівень в ієрархії каталогів. В RedHat стартові скрипти лежать в `/etc/rc.d/init.d/`, а символьні посилання – в каталогах `/etc/rc.d/rcN.d`, але мають такий же вигляд: `S21nfs -> ../init.d/nfs`.

Мандрейк [6] також підтримує структуру в стилі RedHat(`/etc/rc.d/init.d/`), але також і традиційну структуру стилю Юнікс/Дебіан (`/etc/init.d`). В Мандрейк'у це забезпечується символьним посиланням: `/etc/rcN.d -> /etc/rc.d/rcN.d`.

---

[3] На даний час (липень, 2002) вже доступна для завантаження і установки версія 3 Debian'a - Woody. Вона все ще вважається розробниками нестабільною. Але з числених коментарів нестабільний Дебіан – значно стабільніший за стабільні версії багатьох комерційних укладань, особливо таких, як RedHat чи Мандрейк

[4] англ. firewall

[5] В залежності від конкретної реалізації системи. Наприклад, в Лінаксах більшістю прийняті символьні посилання, в той час як Соляріс має жорсткі посилання

[6] Оскільки він починав, як RedHat клон.

3 З'єднання з ISP

Цьому питанню присвячено досить багато літератури і я не буду на ньому зупинятися надовго. Достатньо інформації для старту може надати Linux PPP HOWTO [LWD].

До стандартних скриптів `/etc/ppp/ip-up` (це скрипт, який виконується після установки з'єднання на інтерфейсі ppp0) треба додати кілька рядків, для того, щоб черга повідомлень sendmail очищалась при кожному з'єднанні з інтернетом і читались нові листи, які надійшли на вашу адресу на POP-сервері на інтернет-провайдері.

Для sendmail треба додати такий рядок:

```
/usr/sbin/sendmail -q
```

Всі інші частини установки досить таки стандартні:

3.1 PPPD

Просто наведені кілька файлів конфіґурації без надлишкових коментарів:

Файл /etc/ppp/options

```
lock
nodetach
connect /etc/ppp/asahi-net.chat
defaultroute
idle 300
noauth
modem
debug
noipdefault
user user-kvlv
/dev/ttyS0 115200
```

Файл /etc/ppp/asahi-net.chat

```
/usr/sbin/chat -v \
-r /var/log/\${0}.log \
ABORT BUSY \
```

```
ABORT BLACKLISTED \
ABORT  'NO DIALTONE' \
ABORT 'NO CARRIER' '' \
ATZ OK \
atdp123456789 CONNECT
```

3.2 Демон керування з'єднанням connectd

Для керування з'єднанням я користуюсь пакетом connectd. Можна користуватись і більш розповсюдженим diald [Sch]. Мій вибір зупинився саме на connectd [?], який відрізняється від diald тим, що він сам з своєї ініціативи ніколи не відкриває з'єднання із зовнішньою мережею. Користувач повинен явно відкрити з'єднання з мережею командою

```
# connect open
#
```

. Для домашньої конфіґурації це виявляється зручнішим, оскільки система не намагається зв'язатися з тенетами для кожного запиту до DNS .

Для конфіґурації connectd до файлів pppd потрібно додати такі файли:

Файл /etc/ppp/ppp-on

```
#!/bin/sh
  exec pppd
```

Файл /etc/ppp/ppp-off

```
#!/bin/sh
  killall pppd
```

Файл /etc/ppp/ip-up

```
#!/bin/bash

# This file should not be modified -- make local changes to
# /etc/ppp/ip-up.local instead
```

```
LOGDEVICE=$6
REALDEVICE=$1

### for connectd to understand
kill -SIGUSR1 ‘cat /var/run/connectd.pid‘

#### set time
rdate -s ‘cat /etc/timeservers‘

### clear mail queue
/usr/sbin/sendmail -q


export PATH=/sbin:/usr/sbin:/bin:/usr/bin

echo "$REALDEVICE" > /var/run/ppp-$LOGDEVICE.dev
[ -x /etc/ppp/ip-up.local ] && /etc/ppp/ip-up.local $*

#/etc/sysconfig/network-scripts/ifup-post ifcfg-${LOGDEVICE}
exit 0
```

4 З'єднання з тенетами

В цьому розділі розглядається питання з'єднання з тенетами комп'ютерів локальної мережі. Щоб пакети TCP/IP могли проходити з внутрішньої мережі в зовнішній світ (всесвітні тенета), потрібно мати настроєними як клієнти, так і сервер. Клієнти повинні знати,

- який комп'ютер в локальній мережі служить в якості шлюзу [7] і
- який мережевий інтерфейс [8] клієнта зможе передати пакети клієнта на шлюз і назад.

.

[7] Gateway
[8] Gateway device

Сервери, що виконують функцію шлюза, мають щонайменше два мережевих інтерфейси. [9]

- На сервері в першу чергу повинен бути настроєний режим пересилки пакетів між мережевими інтерфейсами[10].
- Оскільки для описаної конфіґурації використовуються IP-адреси приватного діапазону, вони не повинні попасти в зовнішній світ. Тому, на серверах застосовується техніка переписування мережевих адрес, яка в залежності від реалізації носить назву або IP masquerading [11] [ipm] або Network Address Translation [12] [natb].
- Додатково для під'єднаного до тенет сервера важливим питанням є забезпечення безпеки мережевого сполучення. Питання налаштування брандмауера [13] докладніше розгладається в розділі 6 ("Безпека").

4.1 Шлюз (клієнти)

Для клієнтів на базі Мандрейк/RedHat і подібних для цього пересвідчіться, що файл конфіґурації інтерфейсу мережі (`/etc/sysconfig/network-scripts/ifcfg-eth0` для інтерфейсу `eth0`) містить рядок `GATEWAY`:

Файл /etc/sysconfig/network-scripts/ifcfg-eth0:

```
DEVICE=eth0
ONBOOT=yes
BOOTPROTO=static
IPADDR=192.168.2.10
NETMASK=255.255.255.0
HOSTNAME=myhost
GATEWAY=192.168.2.1
```

Якщо ваші клієнти побудовані на базі Дебіан'а, це встановлюється в файлі `/etc/network/interfaces`:

[9] Інколи обидва інтерфейси фізично розділені. Як, наприклад в розглянутих конфіґураціях на рис. 2.2.1 чи рис. 2.2.2. Інколи на сервері створюється тільки віртуальний інтерфейс на фізичному носії. Як, наприклад, у конфіґурації на рис. 2.2.2. Але, як в тому, так і в іншому випадку потрібно налаштувати сервер для передачі пакетів між двома мережевими інтерфейсами.

[10] IP forwading

[11] маскування IP адрес

[12] Переписування мережевих адрес

[13] Firewall

Файл /etc/network/interfaces:

```
iface eth1 inet static
      address 192.168.2.10
      network 192.168.2.0
      netmask 255.255.255.0
      broadcast 192.168.2.255
      gateway 192.168.2.1
```

4.2 Шлюз (сервер)

На сервері потрібно настроїти дві речі: пересилку пакетів між інтерфейсами ([ipf]) і механізм переписування мережевих адрес. Докладно це описано трохи далі в розділі "Захист доступу до системи". Тут – кілька слів про відмінності IPFW і NAT в світі Лінакс і BSD.

Лінакс-системи і *BSD похідні системи трохи відрізняються в підході до цього питання. Традиційно в усіх Лінакс-системах функція пересилки пакетів і переписування адрес поєднувалась і виконувалась одним програмним модулем. Так було з `ipfw`, `ipchains` [14] і так є з `ip-tables`. З іншого боку в багатьох Юнікс системах ці функції розділені. В MacOS X, і кількох вільних варіантах *BSD (FreeBSD, OpenBSD, NetBSD) ці дві функції також виконуються двома різними програмами (`ipfw` і `natd` відповідно).

Лінакс: На лінакс-системах вмикнути переправлення пакетів можна виконавши команду:

```
# echo 1 > /proc/sys/net/ipv4/ip_forward
#
```

Саме ця команда вказана в скрипті для IP Chains далі по тексту на стор. **??**. Альтернативно можна відредагувати файл в якому встановлюється цей параметр. На Дебіан'і це буде файл `/etc/network/options`. В цьому файлі змініть `ipforward=no` на `ipforward=yes`. В RedHat-подібних системах відредагуйте файл `/etc/sysctl.conf` і змініть `net.ipv4.ipforward=0` на `net.ipv4.ipforward=1`.

[14] `ipfw` використовувався з системами на базі ядер 1.x і 2.0.x. Після цього, з ядрами 2.2.x стало використовуватись `IP Chains` і вже з ядром 2.4.x можна користуватись як `IP Chains` так і `IP Tables`.

Механіка переписування адрес вмикається додатковим параметром "MASQ"до фукнцій пакету IP Chains. Про це теж далі по тексту, в розділі 6 "Захист доступу до системи".

MacOS X: Щоб вмикнути пересилку пакетів між інтерфейсами в сервері на базі MacOS X відредагуйте файл `/etc/hostconfig` і змініть в ньому рядок `IPFORWARDING=-NO-` на `IPFORWARDING=-YES-`. Після цього перевантажте систему.

Переписування адрес забезпечується програмою natd. Достатньо запустити програму таким чином:

```
# natd -dynamic -interface ppp0
#
```

Докладні інструкції по встановленню `natd` викладені на цій сторінці: [mac].

5 Налагоджування служби розв'язки імен (DNS)

Для більш-менш нормальної роботи непогано б мати локальний сервер DNS. Сервер DNS прискорить роботу на повільному модемному зв'язку з тенетами. Не завадить його мати і при швидкому зв'язку з інтернетом, оскільки локальний сервер створює кеш. Локальний кеш прискорює розв'язку часто вживаних назв. Всі комп'ютери на локальній мережі заносяться в DNS і це теж полегшує адміністрування систем. Непотрібно потім додавати ново-збудовані машини до `/etc/hosts` на всіх своїх системах.

Тобто, я вважаю, що я вже вас вмовив – DNS вам потрібен. Тепер перейдем до конкретного конфіґурування.

Для Debian'а є невелике зауваження: яке було мені варте кількох добрячих днів конфіґурування named, переконфіґурування, і таке інше.

Нічого не працювало. Це все при тому, що я просто скопіював свої старі файли з робочої машини. Моїм попереднім сервером DNS був Sparc RedHat 5.2. Аж поки не виявилось, що причина всих моїх страждань до смішного проста. Потрібно було просто уважніше читати Debian-специфічну інформацію в HOWTO [LN+] (одне-єдине речення, на яке зразу-ж звернув увагу тільки-но розібрався в чому проблема).

Будь-яка стандартна книжка чи посібник, HOWTO, ЧаП (Часті Питаня) по DNS вам скаже, що файли конфіґурації bind версії 8.x лежать так: головний файл `/etc/named.conf`, всі інші файли в каталозі `/var/named`. У Debian'і змінене стандартне розташування. Тож, debian'івський bind шукає головний файл конфіґурації в каталозі `/etc/bind`, тобто основним файлом буде `/etc/bind/named`. Далі все залежить тільки від конфіґурації, закладеної в `/etc/bind/named`. Розташування всіх файлів водному каталозі в `/etc/bind` може в майбутньому трохи полегшити вам життя, коли Ви вирішите настроїти named для роботи в середовищі chroot.

5.1 /etc/bind/named.conf

Мій основний конфіґураційний файл з кількома додатковими коментарями:

Файл /etc/bind/named.conf:

```
options {
        directory "/var/named";
// Каталог в якому лежать всі файли зон DNS
//
        /*
         * If there is a firewall between you and nameservers you want
         * to talk to, you might need to uncomment the query-source
         * directive below.  Previous versions of BIND always asked
         * questions using port 53, but BIND 8.1 uses an unprivileged
         * port by default.
         */
         // query-source address * port 53;

 // Тут бажано вказати сервери DNS вашого провайдера. В цьому
 // випадку нерозв'язані локально адреси будуть автоматично
 // розв'язуватись через DNS провайдера, замість того, щоб
 // шукати через зону root

        forward first;
        forwarders {
                1.2.3.4;
                5.6.7.8;
```

```
        };
};

//
// a caching only nameserver config
//
zone "." {
        type hint;
        file "root.hints";
};

zone "0.0.127.in-addr.arpa" {
        type master;
        file "local/127.0.0";
};

 // local zone
 /*
   Тут власне починається конфігурація локальної зони. Для свого
   домашнього домену я вибрав назву ‘sakae’ і адреси з приватного
   діапазону 192.168.2.0 (мережа класу C).
 */

zone "sakae" {
        notify no;
        type master;
        file "sakae/sakae";
};

 /*
    зона для зворотньої розв'язки назв - назву хоста за його IP
    адресою
 */
zone "2.168.192.in-addr.arpa" {
        notify no;
        type master;
        file "sakae/2.168.192";
};
```

5.2 Файли локальної зони

Далі йдуть два файли, що конфіґурують локальну зону. Вона в мене носить назву ‘sakae’.

5.2.1 Файл прямої розв’язки:

Перший для прямої розв’язки (*gethostbyname(3)*):

Файл /var/named/sakae/sakae:

```
; our local zone
$TTL 3D
@       IN      SOA         ns.sakae. hostmaster.sakae. (
                200109231
                8H      ; refresh
                2H      ; retry
                4W      ; expire
                1D )    ; minimum
;
        NS      ns      ; nameserver
        MX      10      mail.sakae.

localhost       A       127.0.0.1

ns              A       192.168.2.1
mail            A       192.168.2.1

yarylo          A       192.168.2.1
                HINFO   "ibm 570" "Mandrake linux"
;
berkut          A       192.168.2.10
                HINFO   "Compaq ProLinea 4/50" "SuSE linux"
;
veles           A       192.168.2.15
                HINFO   "Sparc Station 5" "RH Linux 5.2 sparc"
;
```

```
perun           A       192.168.2.16
                HINFO   "IBM Think Pad 570e" "Mandrake 7.2"
;
mavka           A       192.168.2.20
                HINFO   "PowerPC Macintosh" "Mac OS/Linux"
;
prosha          A       192.168.2.30
                HINFO   "Performa 5220 Macintosh" "Mac OS"
;
natalya         A       192.168.2.70
                HINFO   "Pentium" "Mandrake 8.0"
nfs             CNAME   natalya
fetchmail       CNAME   natalya
```

Невеличкий коментар до наведеного файлу. Домашня мережа виявилась на диво динамічною. Я перебудовую свої системи досить часто в залежності від потреб і від величезного набору факторів: переповнюється диск з домашніми директоріями, і наступає час перебудови сервера NFS; один модем змінює інший (зовнішній 28k під'єднаний до sparc'у змінюється на 56k winmodem у PC) і змінюється сервер, що забезпечує зв'язок з зовнішнім світом, і таке інше. Вся мережа традиційно керується цілою системою розрізнених командних і Перл скриптів часто не зв'язаних між собою. Спочатку я користувався назвами хостів в таких скриптах і, отже, при кожній зміні в мережі, з появою чи відставкою того чи іншого сервера мені було потрібно відшукувати скрипти, які перестали від такої зміни працювати і лагодити їх. Поступово я перейшов до, так би мовити, "функціональних"назв. Тобто, в DNS я став додавати псевда до назв хостів, які виражали б сервіс, що працює на даному хості. Мовою DNS це виражається в CNAME рядках.

Як, наприклад, хост natalya є одночасно сервером домашніх директорій, і тому він також зветься 'nfs'. На цьому ж сервері в мене працює fetchmail [fet], тому у мене він також зветься і fetchmail. Ось записи, що стосуються сервера natalya:

```
natalya         A       192.168.2.70
                HINFO   "Pentium" "Mandrake 8.0"
nfs             CNAME   natalya
fetchmail       CNAME   natalya
```

Всі скрипти я поступово виправляю і записую в них "функціональні псевда"замість справжніх назв. Таким чином, при черговій зміні в домашній мережі мені не потрібно більше виправляти скрипти, а просто досить змінити CNAME в DNS.

5.2.2 Зворотня розв'язка

І файл для зворотньої розв'язки (*gethostbyaddr(3)*). Увага, зважайте на крапки в кінці рядків – вони обов'язкові в цьому файлі. Відсутність крапок - найрозповсюдженіша помилка при конфіґуруванні DNS.

Файл /var/named/sakae/2.168.192:

```
$TTL 3D
@       IN      SOA     ns.sakae. hostmaser.sakae.  (
                                        1       ; Serial
                                        8H      ; Refresh
                                        2H      ; Retry
                                        1W      ; Expire
                                        1D)    ; Minimum
         IN      NS      ns.sakae.com.
1        IN      PTR     yarylo.sakae.
10       IN      PTR     berkut.sakae.
15       IN      PTR     veles.sakae.
16       IN      PTR     perun.sakae.
20       IN      PTR     mavka.sakae.
70       IN      PTR     natalya.sakae.
```

5.2.3 /etc/resolv.conf

Конфіґурування DNS закінчується установкою файлів `/etc/resolv.conf` на сервері та на клієнтах. На сервері це буде таке:

Файл /etc/resolv.conf (сервер):

```
domain sakae
nameserver 127.0.0.1
```

На всіх клієнтах встановлюється такий файл:

Файл /etc/resolv.conf (клієнт):

```
domain sakae
nameserver 192.168.2.1
```

6 Захист доступу до системи.

6.1 Firewall або брандмауер.

6.1.1 Лінакс: IP Chains

Головне джерело інформації при установці IP Chains - це IP Chains HOWTO [Rus]. Найпростіший варіант наведений для прикладу в цьому документі працює без проблем. Якщо вам потрібне щось більш серйозне, зверніться до першоджерела. Ось рекомендації прямо з [Rus]:

> 3.1 Rusty's Three-Line Guide To Masquerading
>
> This assumes that your external interface is called 'ppp0'. Use ifconfig to find out, and adjust to taste.

```
\label{ipforward}
       # ipchains -P forward DENY
       # ipchains -A forward -i ppp0 -j MASQ
       # echo 1 > /proc/sys/net/ipv4/ip_forward
```

Ці три команди встановлюють політику MASQ (маскування) всіх пакетів, що проходять через інтерфейс `ppp0` і дозволяють також `ip_forward` в системі. Після цього достатньо записати встановлені правила в файл командою:

```
# ipchains-save > /etc/ipchains.rules
#
```

І додати до стартових скриптів системи ось цей невеличкий скрипт:

Файл /etc/init.d/packetfilter:

```
#!/bin/sh
# Script to control packet filtering.

# If no rules, do nothing.
[ -f /etc/ipchains.rules ] || exit 0

case "$1" in
    start)
        echo -n "Turning on packet filtering:"
        /sbin/ipchains-restore < /etc/ipchains.rules || exit 1
        echo 1 > /proc/sys/net/ipv4/ip_forward
        echo "."
        ;;
    stop)
        echo -n "Turning off packet filtering:"
        echo 0 > /proc/sys/net/ipv4/ip_forward
        /sbin/ipchains -F
        /sbin/ipchains -X
        /sbin/ipchains -P input ACCEPT
        /sbin/ipchains -P output ACCEPT
        /sbin/ipchains -P forward ACCEPT
        echo "."
        ;;
    *)
        echo "Usage: /etc/init.d/packetfilter {start|stop}"
        exit 1
        ;;
esac

exit 0
```

А також створити на нього посилання:

```
# ln -s /etc/init.d/packetfilter /etc/rcS.d/S39packetfilter
#
```

6.1.2 ipfw в MacOS X.

[далі буде]

6.2 TCP wrappers (файли /etc/hosts.allow,deny)

6.2.1 Клієнт

Спочатку про клієнт, оскільки це простіше.

hosts.deny `/etc/hosts.deny` на клієнті (про сервер - далі) має всього один рядок:

```
ALL: ALL
```

Цей рядок забороняє з'єднання до даного хоста на будь-якому порту з будь-якого місця. Далі спрацьовує файл /etc/hosts.allow, який відкриває з'єднання тільки для тих служб (або для тих хостів), які нам потрібні.

hosts.allow На клієнті `/etc/hosts.allow` має такі рядки:

```
ALL: .sakae
ALL: 127.0.0.1
```

6.2.2 Сервер

inetd та xinted на сервері На сервері, після кількох різних модифікацій і спроб, я зупинився на такому варіанті: сервер стартує sshd і це є єдиним можливим способом отримати командну оболонку на сервері.

Тобто, у встановленні безпеки я пішов параноїдальним шляхом: вбив inetd і прибрав стартові скрипти для нього. Справжній параноїк може навіть стерти сам двійковий файл inetd. [15] Отже, файли `/etc/hosts.*` виявилися незадіяними, тому проблем з їх конфіґурацією на сервері у вас не виникне.

[15] "Параноїк"на мою думку є синонімом до "гарний системний адміністратор"

ssh та sshd Для sshd я додав невеликий скрипт, який стартує sshd при завантаженні системи. Після цього sshd слухає на 22 порті. Також потрібно додати відповідні символьні посилання на стартовий скрипт. Можливо такий скрипт вже буде включено в вашу систему. Наприклад, Мандрейк 8.2 вже має такий скрипт, який встановлюється разом з RPM'ом sshd.

Файл /etc/init.d/sshd:

```
#!/bin/sh
SSHD_PATH=/usr/local/sbin/sshd
case $1 in
        start)
        [ -x ${SSHD} ] && { \
                printf "Starting sshd..." ; \
                exec ${SSHD}
                [ $? = 0 ] && {  echo "OK"; }
                }
                ;;
        stop)
                printf "Stopping sshd..."
                killall sshd;  echo "OK";;
        *)  echo "Usage $(basename $0) start|stop" ;;
esac
```

В Debian'і модифікація стартових скриптів для сервісів здійснюється командою update-rc.d. Щоб створити символьні посилання на наведений вище скрипт виконайте команду

```
# update-rc.d sshd 99
#
```

Це створить посилання `S99sshd` та `K99sshd` в усіх директоріях `/etc/rc?.d`.

6.2.3 Додаткові сервіси

На сервері додатково до вже переліченого бажано заборонити такі сервіси: lpd, portmap, nfs-common, nfs-server, якщо вони не використовуються. Вони неявним чином встановлені і працюють на Debian системі. Потрібно прибрати символьні посилання за `/etc/rc?.d/* -> /etc/init.d/*` і вбити відповідні демони (або перевантажити систему). За допомогою вже згадуваного `update-rc.d` символьні посилання прибираються так:

```
# update-rc.d -f remove сервіс
#
```

7 Конфігурація електронної пошти - сервер та клієнт (sendmail)

Для електронної пошти я вирішив зупинитись на класичному sendmail. Незважаючи на поширену думку про складність, а то навіть і неможливість конфіґурування sendmil'а, я все-таки віддаю перевагу стабільності та надійності "класики". Крім того, освоївши один раз відлагодження програми його потім можна повторювати несчислену кількість разів на інших системах, користуючись одними і тими ж вихідними файлами та методикою. Та й самі вихідні файли не нстільки страшні, як це видається новачку.

Для нашої конфіґурації потрібно буде мати два принципово відмінних конфіґураційних файли для sendmail'а. Перший буде застосовуватись на sendmail сервері, і має деякий "розум"в обробці вхідної, а особливо вихідної пошти. Другий тип – клієнт, який не володіє ніякими особливими розумовими здібностями і просто передає вхідну пошту локальним користувачам, а вихідну – на центральний сервер.

Для настроювання sendmail'а використовується препроцесор `m4`. Вихідні тексти файлів `sendmail.mc`, які служать для ґенерування власне файлів конфіґурації `sendmail.cf` подані далі. Сама ґенерація `sendmail.cf` на Debian'і і у Мандрейку відрізняються, тому далі поданий короткий опис самої процедури.

Після зміни конфіґурації sendamail не забудьте перезапустити сервер, щоб зміни ввійшли в дію:

```
# /etc/init.d/sendmail restart
#
```

7.1 Debian

Крім самого пакету sendmail в Debian'і не потрібно нічого довстановлювати. Всі необхідні файли потрібні для `sendmail.cf` і для `m4` вже входять в сам пакет sendmail. Тож, єдине, що потрібно, це скопіювати `sendmail.mc` в каталог `/etc/mail`, або відредагувати його на місці і виконати таке:

```
# cd /etc/mail; make sendmail.cf
#
```

7.2 Мандрейк

В Мандрейку недостатньо встановити тільки `sendmail-*.rpm` Всі додаткові файли, що служать для настроювання sendmail включені в RPM `sendmail-cf`. Тож його потрібно встановити також. В цьому RPM файл Makefile кладеться в директорію `/usr/lib/sendmail-cf/cf`. І сам Makefile відрізняється від того, що в Debian'і. І, чесно кажучи, з погляду на нього мені цей Makefile здався досить таки непотрібним. А коментар з цього ж файлу тільки потвердив мене в цій думці:

```
#  Create configuration files using "m4 ../m4/cf.m4 file.mc > file.cf";
#  this may be easier than tweaking the Makefile.  You do need to
```

Тому в Мандрейку легше просто скористуватись безпосередньо командою m4 з командної оболонки:

```
# cd /usr/lib/sendmail-cf/cf
# m4 ../m4/cf.m4 [шлях\_до\_файлу\_sendmail.mc] > /etc/mail/sendmail.cf
#
```

7.3 Центральний сервер sendmail

7.3.1 /etc/mail/sendmail.cf

Далі повністю наведений файл sendmail.mc для головного сервера sendmail. Основні функції закладені в конфіґурації такі:

1. використання Smart Host;

2. маскування назв хостів всередині локальної мережі;
3. маскування поштових адрес користувачів у вихідних листах (genericstable).

Файл /etc/mail/sendmail.cf для сервера:

```
dnl# WORKING CONFIG - local mailer procmail /usr/local/bin/procmail
dnl# sendmail 8.9.3
dnl# -------------- DK may 6 2000 -----------
dnl# Configuration file for external mail gateway
dnl# mascarades itself as tokyo.email.ne.jp
dnl#  translataes users with the use of generistable file
dnl#
dnl# needs 2 files: /etc/mail/genericsdomain and /etc/mail/genericstable
dnl#
include(/usr/lib/sendmail-cf/m4/cf.m4)
VERSIONID (‘sendmail.mc - dk’)
OSTYPE(linux)
define(‘ALIAS_FILE’,‘/etc/mail/aliases’)
FEATURE(masquerade_envelope)dnl
FEATURE(‘allmasquerade’)dnl
FEATURE(‘relay_entire_domain’)
FEATURE(genericstable, ‘hash -o /etc/mail/genericstable’)
define(‘SMART_HOST’,‘smtpmail.at-your-provider.jp’)
MASQUERADE_AS(tokyo.email.ne.jp)
MASQUERADE_DOMAIN(yarylo.sakae,yarylo,sakae)
FEATURE(local_procmail)
MAILER(smtp)
```

Далі - по трошку про кожен окремий пункт.

Smart_Host *Smart_Host* взагалі-то для центрального сервера річ не завжди обов'язкова. Але можуть бути випадки, коли без нього робота вашого поштового сервера може виявитись або неможливою, або близькою до неможливості. Я довгий час не мав *Smart_Host*'а і працював нормально (загалом), але коли додав це до конфіґурації, відчув різницю в швидкості відправки пошти особливо для деяких адрес.

Суть ось у чому. Без *Smart_Host*'а ваш поштовий сервер для відправки листа адресату зв'язується з 25 портом того хоста, який вказаний в заголовку *"To: "*

листа (або, якщо бути точним, з сервером, що вказаний в MX-полі, що відповідає хосту з *"To: "*). Інколи це працює нормально (якщо ваш респондент знаходиться близько від вас – в поняттях мережі, тобто зв'язок з ним швидкий). Але для деяких адресатів затримка в передачі пошти досить таки суттєва. Крім того, ваш провайдер може просто перекрити вихід в зовнішню мережу через 25 порт.

В такому випадку потрібно скористатись параметром *Smart_ Host* для sendmail. Один рядок в sendmail.mc файлі вмикає цю опцію:

```
define(‘SMART\_HOST’,‘smtpmail.at-your-provider.jp’)
```

Ваш провайдер інтернет послуг повинен був вам повідомити назву SMTP сервера, яким Ви маєте користуватись для передачі пошти. Назву сервера запишіть в цьому рядку.

Маскування внутрішнього домену та імен користувачів (masquerading, genericstable)
Якщо, скажімо комп'ютер у внутнішній мережі носить назву yarylo, локальна зона DNS названа sakae, а ваше ім'я користувача на цій системі є 'dk', то всі вихідні листи будуть доходити до ваших адресатів із записом '`dk@yarylo.sakae`' в заголовку *'From:'*. Натиснувши Reply чи Відповісти в своєму поштовому клієнті ваш респондент відправить листа саме по цій адресі, якої насправді не існує в інтернеті.

Щоб цього не траплялось потрібно щоб адреси при виході із внутрішньої мережі в зовнішній світ переписувались відповідним чином. Цю проблему можна також вирішити певною настройкою поштового клієнта, але це має свої недоліки. Наприклад, в деяких клієнтах можна встановити *"From:"* поле ще при відправці пошти. Тобто, якщо моя поштова адреса на POP сервері провайдера є `kov@tokyo.email.ne.jp`, я можу, наприклад, в exmh вставити додатковий рядок

```
From: kov@tokyo.email.ne.jp
```

і цим проблема буде вирішена для всіх адресатів, що знаходяться назовні. Але, якщо я захочу переслати листа всередині моєї внутрішньої мережі, відповідь мені все-таки повинна буде йти через інтернет.

Крім того такі установки потрібно буде робити для всіх користувачів окремо і окремо для всіх поштових клієнтів, які ці користувачі вживають. Тому краще буде виправити це на рівні MTA один раз і назавжди.

Переписування адрес насправді складається з двох частин: маскування внутрішньог домену, та маскування адрес користувачів.

Маскування внутрішнього домену Для цього потрібно просто сказати sendmail'у яка назва вашого домену, і як подавати комп'ютери цього домену назовні. Для цього служать подані далі рядки в конфіґурації sendmail. Змініть їх відповідно до назв ваших систем та вашої справжньої адреси на інтернет-провайдері.

```
FEATURE(masquerade_envelope)dnl
FEATURE(‘allmasquerade’)dnl
FEATURE(‘relay_entire_domain’)
MASQUERADE_AS(tokyo.email.ne.jp)
MASQUERADE_DOMAIN(yarylo.sakae,yarylo,sakae)
```

Переписування імен користувачів Для цього Вам потрібно скористуватися опцією *Generics Table* в конфіґурації sendmail. Вкажіть sendmail'у, щоб він користувався при переписуванні адрес базою даних *Generics Table*. Це визначається таким рядком в sendmail.mc:

```
FEATURE(genericstable, ‘hash -o /etc/mail/genericstable’)
```

Після цього створіть власне саму базу даних. Файл `/etc/mail/genericstable` містить всі адреси внутрішніх користувачів з їхніми відповідними адресами в інтернеті. В моєму конкретному прикладі цей файл складається з двох рядків - для мене і моєї дружини:

Файл /etc/mail/genericstable:

```
dk kov@tokyo.email.ne.jp
nk kov@tokyo.email.ne.jp
```

Поля в файлі розділяються просто звичайними пропусками. Після того, як такий файл створено його потрібно перетворити в базу. Для цього виконайте:

```
# cd /etc/mail
# makemap hash genericstable
#
```

Це створить в каталозі `/etc/mail` файл `genericstable.db`.

7.4 Клієнт sendmail

Єдиною опцією якою користується клієнт sendmail є *Smart_ Host*. Тому його файл `sendmail.mc` досить простий. В якості *Smart_ Host*'а використовується центральний поштовий сервер в локальній мережі.

Файл sendmail.cf для клієнтів: `sendmail.cf-client`

7.5 fetchmail

[далі буде]

7.6 Деякі інші проблеми з sendmail.

Під час налагоджування своїх машин я зіткнувся з кількома маленькими проблемками. Можливо не всі ці проблемки я згадав, але ось те, що мені запам'яталось.

7.6.1 Неявно встановлений поштовий сервер

В стандартних конфіґураціях поштовими серверами Debian'a та Мандрейка є exim та postfix відповідно. Тож, перш, ніж приступати до настроювання sendmail потрібно забрати з диску, встановлені там exim чи postfix і встановити натомість sendmail. `'rpm -e'` та `'dselect'` – ваші друзі.

7.6.2 На M7.2 не працює відсилання пошти для будь-кого крім root.

Стандартний RPM пакет sendmail на M7.2 дає таку помилку для кожного користувача, який відсилає листа:

```
Can't create transcript file ./xff8R6lwP18368: Permission denied
```

Я не зустрічав такої помилки на M8.0. Можливо вона вже виправлена. Це повідомлення з'являється для кожного користувача крім root. Проблема в тому, що власники двійкового файлу `/usr/sbin/sendmail` і директорії черг не співпадають. Крім того sendmail має працювати з SUID, SGID. Вихід з цього становища такий:

```
# chown bin:mail /var/spool/mqueue
# chmod +s /usr/sbin/sendmail
#
```

7.6.3 Мандрейк 8.0 не дає перебудувати поштові псевда (/etc/mail/aliases).

Нормальна (безпечна) конфіґурація sendmail не дозволяє користуватися умовними посиланнями для файлів бази даних поштових псевд. В Мандрейку (оскільки неявним поштовим сервером вибраний postfix) існує файл `/etc/aliases.db`, який є посиланням на `/etc/mail/aliases.db`. При виконанні команди newaliases з'являється повідомлення про недопустимість символьних посилань:

```
# newaliases
hash map "Alias0": unsafe map file /etc/aliases.db: Symbolic links not allowe
WARNING: cannot open alias database /etc/aliases
Cannot create database for alias file /etc/aliases
```

Вирішення цієї проблеми: витерти символьне посилання і перезапустити команду newaliases.

```
# rm /etc/aliases.db
# newaliases
/etc/aliases: 14 aliases, longest 10 bytes, 152 bytes total
#
```

8 Маленькі поради

8.1 Псевдо-статична IP адреса сервера

[далі буде]

8.2 Безпечний тунель через httptunell та ssh.

[далі буде]

9 Версія цього документу

- 9 липня 2002 р.- остання (поточна) версія

- Thu Nov 22 20:53:24 JST 2001 - перший реліз

- 1.4a Tue Jul 9 15:59:52 JST 2002

10 Історія змін до документу

- Revision 1.4a - вичитані кілька розділів. Можна вважати перша-бета.
- Revision 1.4 2002/04/03 08:31:51 dk99034 - Форматування тексту вичищено, щоб з нього можна було скриптом зробити більш-менш пристойний html

  - виправленя, стилістичні і граматичні.
- Revision 1.3 2002/04/02 06:32:54 dk99034 - розділи про фізичну конфіґурацію з'єднань (три різні схеми - модем, ADLS з 1 мережевою платою, ADLS з 2 платами);

  - розділи про настроювання клієнтів - лінакс/макос;

  - вставлені заготовки розділів "поради"no-ip, httptunnel

- TODO - подописувати розділи "поради"і ipfw для macosx.

Література

[con] *Internet Connection Sharing, and dialup management software.* http://freespace.virgin.net/fuchsia.groan/software/index.html.

[deb] http://www.debian.org/.

[fet] http://www.tuxedo.org/ esr/fetchmail/.

[ipf] http://www.troubleshooters.com/linux/ip_fwd.htm.

[ipm] *Linux IP Masquerade Resource.* http://www.e-infomax.com/ipmasq.

[LN+] Nicolai Langfeldt, Jamie Norrish, et al. *DNS HOWTO.* http://www.linuxdoc.org/HOWTO/DNS-HOWTO.html.

[LWD] Corwin Light-Williams and Joshua Drake. *Linux PPP HOWTO.* http://www.linuxdoc.org/HOWTO/PPP-HOWTO/.

[mac] http://www.macosx.org/nat.html.

[nata] http://www3.sympatico.ca/dccote/appleshareipoverpppoe.html.

[natb] *Network Address Translation FAQ.* http://www.vicomsoft.com/knowledge/reference/nat.html.

[ppp] http://www.roaringpenguin.com/pppoe.

[pri] *Address Allocation for Private Internets.* http://www.cis.ohio-state.edu/cgi-bin/rfc/rfc1918.html.

[Rus] Rusty Russell. *Linux IPCHAINS-HOWTO.* http://www.linuxdoc.org/HOWTO/IPCHAINS-HOWTO.html.

[Sch] Eric Schenk. *The Linux Diald FAQ.* http://diald.unix.ch/FAQ/diald-faq.html.