L220: Advanced Linux System Administration II

course materials

originally released under the GFDL by LinuxIT
modified and released under the GFDL by University of Zagreb University Computing Centre SRCE ("the publisher")
Acknowledgments
The original material was made available by LinuxIT's technical training centre www.linuxit.com.
The original manual is available online at http://savannah.nongnu.org/projects/lpi-manuals/.
The modified version of this manual is available at http://www.srce.unizg.hr/linux-akademija/.

History
CVS version 0.0 January 2004, Adrian Thomasset <adrian@linuxit.com>.
Reviewed/Updated April 2004, Andrew Meredith <andrew@anvil.org>.
Review/Update May 2005, Adrian Thomasset <adriant@linuxit.com>.
February 2014. Title: L220: Advanced Linux System Administration II (version 1.0). Revised and modified at University of Zagreb University Computing Centre SRCE (“the publisher”) by Vladimir Braus.

Notations
Commands and filenames will appear in the text in bold.
The <> symbols are used to indicate a non optional argument.
The [] symbols are used to indicate an optional argument

Commands that can be typed directly in the shell are highlighted as below

```
command
```

No Guarantee
The manual comes with no guarantee at all.
As the major national infrastructural ICT institution in the area of research and higher education in Croatia, the University Computing Centre SRCE is providing a modern, sustainable and reliable e-infrastructure for research and education community.

This includes computing and cloud services, high performance computing, advanced networking, communication systems and services, middleware, data and information systems and infrastructure. At the same time SRCE acts as the computing and information centre of the largest Croatian university – the University of Zagreb, and is responsible for the coordination of the development and usage of e-infrastructure at the University.

Furthermore, by applying cutting edge technologies SRCE continuously enriches academic and research e-infrastructure and its own service portfolio. This enables the active participation of Croatia and Croatian scientists in European and global research and higher education area and projects.

Since its founding in 1971 as a part of the University of Zagreb, at that time the only Croatian university, SRCE has provided an extended advisory and educational support to institutions and individuals from the academic and research community in the use of ICT for education and research purposes.

From its beginnings, and still today, SRCE has been recognized as an important factor of the development of modern e-infrastructure at the national level, participating in different projects and providing services like Croatian Internet eXchange (CIX).

SRCE has a 41 year old tradition of organizing professional courses from the field of ICT.

University Computing Centre SRCE
Josipa Marohnića 5
10000 Zagreb
Croatia

http://www.srce.unizg.hr
e-mail: ured@srce.hr
phone: +385 1 6165 555
# Table of Contents

## DNS

<table>
<thead>
<tr>
<th>Section</th>
<th>Page</th>
</tr>
</thead>
<tbody>
<tr>
<td>1. Basic Bind Configuration</td>
<td>7</td>
</tr>
<tr>
<td>1.1 The Logging Statement</td>
<td>7</td>
</tr>
<tr>
<td>1.2 The Options Statement</td>
<td>8</td>
</tr>
<tr>
<td>1.3 The Zone Statement</td>
<td>10</td>
</tr>
<tr>
<td>1.4 The Access Control Lists (acl) Statement</td>
<td>11</td>
</tr>
<tr>
<td>2. Create and Maintain Zone Files</td>
<td>12</td>
</tr>
<tr>
<td>3. Securing a DNS Server</td>
<td>13</td>
</tr>
<tr>
<td>3.1 Server Authentication</td>
<td>15</td>
</tr>
<tr>
<td>3.2 DATA Integrity and Authenticity</td>
<td>17</td>
</tr>
</tbody>
</table>

## MAIL AND LISTS

<table>
<thead>
<tr>
<th>Section</th>
<th>Page</th>
</tr>
</thead>
<tbody>
<tr>
<td>1. Using Sendmail</td>
<td>19</td>
</tr>
<tr>
<td>1.1 Configuration Settings</td>
<td>19</td>
</tr>
<tr>
<td>1.2 Virtual Hosting</td>
<td>21</td>
</tr>
<tr>
<td>2. Configuring Mailing Lists</td>
<td>22</td>
</tr>
<tr>
<td>2.1 Majordomo and Sendmail</td>
<td>22</td>
</tr>
<tr>
<td>3. Managing Mail Traffic</td>
<td>24</td>
</tr>
<tr>
<td>3.1 Mail Filtering with Procmal</td>
<td>24</td>
</tr>
</tbody>
</table>

## WEB SERVICES

<table>
<thead>
<tr>
<th>Section</th>
<th>Page</th>
</tr>
</thead>
<tbody>
<tr>
<td>1. Implementing a Web Server</td>
<td>27</td>
</tr>
<tr>
<td>1.1 Installing Apache</td>
<td>27</td>
</tr>
<tr>
<td>1.2 Monitoring apache load</td>
<td>27</td>
</tr>
<tr>
<td>1.3 Using Apachectl</td>
<td>28</td>
</tr>
<tr>
<td>1.4 Basic Configuration Options</td>
<td>29</td>
</tr>
<tr>
<td>1.5 Restricting Client Access</td>
<td>31</td>
</tr>
<tr>
<td>1.6 Client Basic Authentication</td>
<td>31</td>
</tr>
<tr>
<td>2. Maintaining a Web Server</td>
<td>32</td>
</tr>
<tr>
<td>2.1 HTTPS Overview</td>
<td>32</td>
</tr>
<tr>
<td>2.2 SSL Virtual Hosts</td>
<td>33</td>
</tr>
<tr>
<td>2.3 Managing Certificates</td>
<td>34</td>
</tr>
<tr>
<td>2.4 Virtual Hosts</td>
<td>35</td>
</tr>
<tr>
<td>3. Implementing a Proxy Server</td>
<td>37</td>
</tr>
<tr>
<td>3.1 Getting Started</td>
<td>37</td>
</tr>
<tr>
<td>3.2 Access Lists and Access Control</td>
<td>37</td>
</tr>
<tr>
<td>3.3 Additional Configuration Options</td>
<td>39</td>
</tr>
<tr>
<td>3.4 Reporting Tools</td>
<td>40</td>
</tr>
<tr>
<td>3.5 User Authentication (using PAM)</td>
<td>42</td>
</tr>
</tbody>
</table>
# Table of Contents

**NETWORK CLIENT MANAGEMENT**
- [1. DHCP Configuration](#1-dhcp-configuration) .................................................. 44
  - 1.1 Default DHCP Configurations ................................................................. 44
  - 1.2 Dynamic DNS ...................................................................................... 46
  - 1.3 DHCP Relay ......................................................................................... 48
- [2. NIS Configuration](#2-nis-configuration) ...................................................... 49
  - 2.1 Master Server Configuration .................................................................. 49
  - 2.2 Slave Server Configuration .................................................................... 50
  - 2.3 Client Setup .......................................................................................... 50
  - 2.4 Setting up NFS home directories ......................................................... 51
  - 2.5 Basic NIS Administration ...................................................................... 51
- [3. LDAP Configuration](#3-ldap-configuration) ................................................. 52
  - 3.1 What is ldap ......................................................................................... 52
  - 3.2 OpenLDAP server configuration .............................................................. 53
  - 3.3 Client configuration files ...................................................................... 54
  - 3.4 Migrating System Files to LDAP ............................................................. 54
  - 3.5 LDAP Authentication Scheme ............................................................... 57
- [4. PAM Authentication](#4-pam-authentication) .................................................. 59
  - 4.1 PAM Aware Applications ....................................................................... 59
  - 4.2 PAM Configuration .............................................................................. 60

**SYSTEM SECURITY**
- [1. Iptables/Ipchains](#1-iptables-ipchains) ......................................................... 61
  - 1.1 The Chains ............................................................................................ 61
  - 1.2 The Tables ............................................................................................ 62
  - 1.3 The Targets ........................................................................................... 62
  - 1.4 Example Rules ..................................................................................... 63
- [2. Differences with Ipchains](#2-differences-with-ipchains) ............................ 64
- [3. Security Tools](#3-security-tools) ................................................................. 65
  - 3.1 SSH ...................................................................................................... 65
  - 3.2 LSOF ..................................................................................................... 67
  - 3.3 NETSTAT ............................................................................................. 68
  - 3.4 TCPDUMP ............................................................................................ 68
  - 3.5 NMAP .................................................................................................. 70

**GNU Free Documentation License** ................................................................... 72
Computer name resolution can be performed in a number of ways, including /etc/hosts file and DNS. /etc/hosts file is a convenient way to manage name resolution for a small number of computers, such as a small home network with just two or three machines. /etc/hosts must be updated on every computer on a network whenever any machine’s name or IP address changes or whenever a computer is added to or removed from the network. In addition to /etc/hosts and DNS, several other name resolution systems exist, including Network Information Service (NIS), Windows Internet Name Service (WINS), and more.

### 1. Basic Bind Configuration

The configuration file for a Bind server is `/etc/named.conf`. This file has the following main entries:

<table>
<thead>
<tr>
<th>Main entries in <code>named.conf</code></th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>logging</td>
<td>Specify where logs are written too and what needs to be logged</td>
</tr>
<tr>
<td>options</td>
<td>Global options are set here (e.g. the path to the zone files)</td>
</tr>
<tr>
<td>zone</td>
<td>Defines a zone: the name, the zone file, the server type</td>
</tr>
<tr>
<td>acl</td>
<td>Access control list</td>
</tr>
<tr>
<td>server</td>
<td>Specific options for remote servers</td>
</tr>
</tbody>
</table>

Let's look at a typical configuration file for a caching only server. We will add entries to it as we go to create new zones, logging facilities, security, etc.
### 1.1 The Logging Statement

The syntax for logging is:

```plaintext
logging {
  channel "channel_name" {
    file "file_name";
    versions number_of_files;
    size log_size;
    syslog < daemon | auth | syslog | authpriv | local0 -to- local7 | null >;
    severity <critical | error | warning | notice | info | debug | dynamic >;
    print-category yes_or_no;
    print-severity yes_or_no;
    print-time yes_or_no;
  }
  category "category_name" {
    "channel_name";
  }
};
```

The **channel** defines where logs are sent to (file, syslog or null). If syslog is selected then the facility and the log level can be specified too.
The **category** clause defines the type of information sent to a given channel (or list of channels). The type of channel is given then the default logging facility is used.

```plaintext
category default { default_syslog; default_debug; };
```

**Example:**

We choose not to use the syslog daemon and log everything to a file called "LOG" that will be created in the same directory as the zone files (default `/var/named/`). For this we will create the channel **foo_channel**. Next we want to log **queries** using this channel.

The entry in **named.conf** will look like this:

```
logging {
    channel foo_channel {
        file "LOG";
        print-time yes;
        print-category yes;
        print-severity yes;
    };
    category "queries" {
        "foo_channel";
    };
};
```

Categories such as **queries** are predefined and listed in the **named.conf(5)** manpages. However some of the names have changed since BIND 8, so we include as a reference the list of categories for BIND 9 below:

### BIND 9 Logging Categories

<table>
<thead>
<tr>
<th>Category</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>default</td>
<td>Category used when no specific channels (log levels, files ...) have been defined</td>
</tr>
<tr>
<td>general</td>
<td>Catch all for messages that haven't been classified below</td>
</tr>
<tr>
<td>database</td>
<td>Messages about the internal zone files</td>
</tr>
<tr>
<td>security</td>
<td>Approval of requests</td>
</tr>
<tr>
<td>config</td>
<td>Processing of the configuration file</td>
</tr>
<tr>
<td>resolver</td>
<td>Information about operations performed by clients</td>
</tr>
<tr>
<td>xfer-in or xfer-out</td>
<td>Received or sent zone files</td>
</tr>
<tr>
<td>notify</td>
<td>Log NOTIFY messages</td>
</tr>
<tr>
<td>client</td>
<td>Client activity</td>
</tr>
<tr>
<td>update</td>
<td>Zone updates</td>
</tr>
<tr>
<td>queries</td>
<td>Client Queries</td>
</tr>
<tr>
<td>dnssec</td>
<td>DNSEC transactions</td>
</tr>
<tr>
<td>lame-servers</td>
<td>Transactions sent from servers marked as lame-servers</td>
</tr>
</tbody>
</table>
1.2 The Options Statement

The global options for the server are set at the beginning of named.conf. The syntax is:

```
options{
    option1;
    option2;
    ....
};
```

We next cover the most common options.

**version**

Manpage says “The version the server should report via the ndc command. The default is the real version number of this server, but some server operators prefer the string (surely you must be joking)”

```
version " (surely you must be joking) ";
```

**directory**

The working directory of the server

```
directory "/var/named";
```

**fetch-glue** (default yes) - obsolete

Prevent the server from resolving NS records (the additional data section). When a record is not present in the cache BIND can determine which servers are authoritative for the newly queried domain. This is often used in conjunction with recursion no.

**notify** (default yes)

Send DNS NOTIFY messages to the slave servers to notify zone changes (helps speed up convergence)

**recursion** (default yes)

The server will perform recursive queries when needed

**forward** (only or first)

The default value is first and causes the server to query the forwarders before attempting to answer a query itself. If the option is set to only the server will always ask the forwarders for an answer. This option has to be used with forwarders.
### forwarders (list)

<table>
<thead>
<tr>
<th>List of servers to be used for forwarding. The default is an empty list.</th>
</tr>
</thead>
<tbody>
<tr>
<td>forwarders { 10.0.0.1; 10.0.0.10;};</td>
</tr>
</tbody>
</table>

### datasize

<table>
<thead>
<tr>
<th>Limit the size of the cache</th>
</tr>
</thead>
<tbody>
<tr>
<td>datasize 512M;</td>
</tr>
</tbody>
</table>

### allow-query (list)

<table>
<thead>
<tr>
<th>A list of hosts or networks that may query the server</th>
</tr>
</thead>
</table>

### allow-recursion (list)

<table>
<thead>
<tr>
<th>List of hosts that can submit recursive queries</th>
</tr>
</thead>
</table>

### allow-transfer (list)

<table>
<thead>
<tr>
<th>List of hosts (usually the slaves) who are allowed to do zone transfers</th>
</tr>
</thead>
</table>

## 1.3 The Zone Statement

The syntax for a zone entry in `named.conf` is as follows:

```plaintext
zone domain_name {
    type zone_type;
    file zone_file;
    local_options;
};
```

We first look at the `local_options` available. Some of these are the same options with the same syntax as the global options we have just covered (with some additional ones). The most common ones are `notify`, `allow-transfer` and `allow-query`. Additional ones are `masters` (list of master servers) or `dialup`.

The `domain_name` is the name of the domain we want to keep records for. For each domain name there is usually an additional zone that controls the local in-addr.arpa zone.

The `zone_type` can either be

- `master` the server has a master copy of the zone file
- `slave` the server has a version of the zone file that was downloaded from a master server
- `hint` predefined zone containing a list of root servers
- `stub` similar to a `slave` server but only keeps the NS records
The `zone_file` is a path to the file containing the zone records. If the path is not an absolute path then the path is taken relatively to the directory given earlier by the `directory` option (usually /var/named).

Example master zone entries, allowing zone transfers to a slave server at 10.1.2.3:

```plaintext
zone seafront.bar {
    type master;
    file "seafront.zone";
    allow-transfer{10.1.2.3};
};
zone 2.1.10.in-addr.arpa {
    type master;
    file "10.1.2.zone";
    allow-transfer{10.1.2.3};
};
```

The next example is the corresponding `named.conf` `zone` section for the slave server, assuming the master has the IP 10.1.2.1:

```plaintext
zone "seafront.bar" IN {
    type slave;
    masters {10.1.2.1};
    file "slave/seafront.zone";
};
zone "2.1.10.in-addr.arpa" IN {
    type slave;
    masters {10.1.2.1};
    file "slave/10.1.2.local";
};
```

### 1.4 The Access Control Lists (acl) Statement

Rather than use IPs it is possible to group lists of IP addresses or networks and assign a name to this grouping.

Example acl:

```plaintext
acl internal_net {10.0.0.0/8; }
```
There are built-in ACLs as follow:

<table>
<thead>
<tr>
<th>ACL</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>any</td>
<td>all hosts</td>
</tr>
<tr>
<td>none</td>
<td>no host</td>
</tr>
<tr>
<td>localhost</td>
<td>all IP address for the local interfaces</td>
</tr>
<tr>
<td>localnets</td>
<td>network associated to the localhost interfaces</td>
</tr>
</tbody>
</table>

The Server Statement

This statement is used to assign configuration options for a specific server. For example if a server is giving bad information it can be marked as **bogus**. One can also set the **keys** associated with a server for hosts **authentication** when using DNSSEC (see section 4. Securing a DNS Server)

2. Create and Maintain Zone Files

The format of the zone files is defined in RFC 1035 and contains resource records (RR) for the administered domain or sub-domain.

The types of resource records are:

1 – Start Of Authority (SOA)

```
root-name TTL IN SOA name-server email-address (  
    serial number;  
    refresh;  
    retry;  
    expire;  
    minimum;  
)
```

The SOA record includes the following details:
- the primary name server for the domain
- the responsible party for the domain
- a timestamp (serial number) that changes whenever you update your domain
- the number of seconds before the zone should be refreshed
- the number of seconds before a failed refresh should be retried
- the upper limit in seconds before a zone is considered no longer authoritative
- the negative result TTL (for example, how long a resolver should consider a negative result for a subdomain to be valid before retrying).
The root-name is often replaced with an “@” symbol which resolves to the name of the zone specified in named.conf.

Example:

```
$TTL 86400
@ 1D IN SOA ns.seafront.bar. root.seafront.bar. ( 46 ; serial (d. adams) 1H ; refresh 15M ; retry 1W ; expiry 1D ) ; minimum
```

2 – Records defining the name-servers for this domain, NS records

domain-name IN NS name-server

Example:

```
IN NS ns
```

**NOTICE**

1. If the name of the domain is missing then @ is assumed
2. The fully qualified name of the name-server is ns.seafront.bar. A host name that doesn’t end with a dot will automatically have the domain-name ‘@’ appended to it. Here for example
   
   `ns` becomes `ns.seafront.bar`.

3 – Records defining the mail-servers for this domain, MX records

`domain-name IN MX PRI mail-server`

The PRI entry is a priority number. If several mail-servers are defined for a domain then the servers with the lowest priority number are used first.

4 – Authoritative information for hosts on the domain, called A records

`host-name IN A IP-address`

**Authority Delegation**

When defining the name-servers responsible for another sub-domain additional NS records are added as well as some glue records which are simple A records resolving the DNS servers.
Example:

devel.myco.com. IN NS ns1.devel.myco.com
ns1 IN A 192.168.21.254

Reverse zone files

5 – Authoritative PTR records, resolving IP addresses

n IN PTR host-name

3. Securing a DNS Server

In 1995, following major security flaws discovered in DNS, a new topic called DNSSEC was started within the IETF. This DNSSEC protocol is described in a sequence of three draft documents known as RFC2535bis and proposes to handle server authentication as well as data authenticity.

3.1 Server Authentication

DNSSEC attempts to handle vulnerabilities that occur during unauthorised dynamic updates as well as spoofed master impersonations. These involve host-to-host authentications between either a DHCP or a slave server and the master server.

The dnssec-keygen tool is used to generate a host key on the master server that can then be transferred on a slave server. This authentication mechanism is call TSIG and stands for Transaction Signature. Another mechanism is SIG0 and is not covered in these notes.

Master Configuration

1. First generate the host key on the master server called seafront.bar:

```
dnssec-keygen -a HMAC-MD5 -b 256 -n host seafront.bar.
```

   This will create the following public and a private key pair:

   Kseafront.bar.+157+49196.key
   Kseafront.bar.+157+49196.private

   Notice: These keys must NOT be inserted in the zone files (there is an IN KEY section in the public key that is misleading, looks like a RR). The public and the private keys are identical: this means that the private key can be kept in any location. This also means that the public key shouldn't be published.
The content of the Kseafront.bar.+157+49196.key is:

```
seafront.bar. IN KEY 512 3 157 QN3vIApnV76WS+a2Hr3qj+AqZjpuPjQgVWeeMMGSBC4=
```

2. In the same directory as the server's `named.conf` configuration file create the file `slave.key` with the following content:

```
key "seafront.bar." {
    algorithm hmac-md5;
    secret "QN3vIApnV76WS+a2Hr3qj+AqZjpuPjQgVWeeMMGSBC4=";
};
```

3. Apply the following changes in `named.conf`:

```
include "/etc/slave.key";

zone "seafront.bar" IN {
    type master;
    file "seafront.zone";
    allow-transfer { key seafront.bar.; };
};

zone 2.1.10.in-addr.arpa {
    type master;
    file "10.1.2.zone";
    allow-transfer { key seafront.bar.; };
};
```

**Slave Configuration**

Copy the `slave.key` file to the slave server in the directory containing `named.conf`. Add the following `server` and `include` statements to `named.conf`:

```
server 10.1.2.1 {
    keys {seafront.bar.;};
};

include "/etc/slave.key";
```

**Troubleshooting**

Restart named on both servers and monitor the logs. Notice that DNSSEC is sensitive to time stamps so you will need to synchronise the servers (using NTP). Then run the following command on the master server in the same directory where the dnssec keys
where generated:

```
dig @10.1.2.1 seafront.bar AXFR -k Kseafront.bar.+157+49196.key
```

### 3.2 DATA Integrity and Authenticity

This aspect of DNSSEC is above the level of this manual and is simply a summary of the concepts involved.

Data authenticity may be compromised at different levels.

The recognised areas are:

- altered slave zone files
- cache impersonation
- cache poisoning.

#### New RR records

The integrity and authenticity of data is guaranteed by signing the Resource Records using a private key. These signatures can be verified using a public DNSKEY. Only the validity of the DNSKEY needs to be established by the parent server or "delegation signer" DS.

So we have the following new RRs in the zone files:

<table>
<thead>
<tr>
<th>RR</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>RRSIG</td>
<td>the signature of the RR set</td>
</tr>
<tr>
<td>DNSKEY</td>
<td>public key used to verify RRSIGs</td>
</tr>
<tr>
<td>DS</td>
<td>the Delegation Signer</td>
</tr>
</tbody>
</table>

#### Signing Zone Records

These are the basic steps:

1. Create a pair of public/private zone signing keys (ZSK)

   ```
   dnssec-keygen -a DSA -b 1024 -n zone seafront.bar.
   ```

   You should get two files such as these:

   ```
   Kseafront.bar.+003+31173.key
   Kseafront.bar.+003+31173.private
   ```
2. Insert the public key into the unsigned zone file:

    cat Kseafront.bar.+003+31173.key >> seafront.bar

3. Sign the zone file

    dnssec-signzone -o seafront.bar Kseafront.bar.+003+31173

You should see a message such as:

```
WARNING WARNING WARNING WARNING WARNING WARNING WARNING WARNING WARNING WARNING WARNING WARNING
WARNING WARNING WARNING WARNING WARNING WARNING WARNING WARNING WARNING WARNING WARNING WARNING
WARNING WARNING WARNING WARNING WARNING WARNING WARNING WARNING WARNING WARNING WARNING WARNING
WARNING WARNING WARNING WARNING WARNING WARNING WARNING WARNING WARNING WARNING WARNING WARNING
WARNING WARNING WARNING WARNING WARNING WARNING WARNING WARNING WARNING WARNING WARNING WARNING
WARNING WARNING WARNING WARNING WARNING WARNING WARNING WARNING WARNING WARNING WARNING WARNING
WARNING WARNING WARNING WARNING WARNING WARNING WARNING WARNING WARNING WARNING WARNING WARNING
WARNING WARNING WARNING WARNING WARNING WARNING WARNING WARNING WARNING WARNING WARNING WARNING
WARNING WARNING WARNING WARNING WARNING WARNING WARNING WARNING WARNING WARNING WARNING WARNING
WARNING WARNING WARNING WARNING WARNING WARNING WARNING WARNING WARNING WARNING WARNING WARNING
WARNING WARNING WARNING WARNING WARNING WARNING WARNING WARNING WARNING WARNING WARNING WARNING
WARNING WARNING WARNING WARNING WARNING WARNING WARNING WARNING WARNING WARNING WARNING WARNING
WARNING WARNING WARNING WARNING WARNING WARNING WARNING WARNING WARNING WARNING WARNING WARNING
WARNING WARNING WARNING WARNING WARNING WARNING WARNING WARNING WARNING WARNING WARNING WARNING
WARNING WARNING WARNING WARNING WARNING WARNING WARNING WARNING WARNING WARNING WARNING WARNING
WARNING WARNING WARNING WARNING WARNING WARNING WARNING WARNING WARNING WARNING WARNING WARNING
WARNING WARNING WARNING WARNING WARNING WARNING WARNING WARNING WARNING WARNING WARNING WARNING
WARNING WARNING WARNING WARNING WARNING WARNING WARNING WARNING WARNING WARNING WARNING WARNING
WARNING WARNING WARNING WARNING WARNING WARNING WARNING WARNING WARNING WARNING WARNING WARNING
WARNING WARNING WARNING WARNING WARNING WARNING WARNING WARNING WARNING WARNING WARNING WARNING
WARNING WARNING WARNING WARNING WARNING WARNING WARNING WARNING WARNING WARNING WARNING WARNING
WARNING WARNING WARNING WARNING WARNING WARNING WARNING WARNING WARNING WARNING WARNING WARNING
WARNING WARNING WARNING WARNING WARNING WARNING WARNING WARNING WARNING WARNING WARNING WARNING
WARNING WARNING WARNING WARNING WARNING WARNING WARNING WARNING WARNING WARNING WARNING WARNING
WARNING WARNING WARNING WARNING WARNING WARNING WARNING WARNING WARNING WARNING WARNING WARNING
WARNING WARNING WARNING WARNING WARNING WARNING WARNING WARNING WARNING WARNING WARNING WARNING
WARNING WARNING WARNING WARNING WARNING WARNING WARNING WARNING WARNING WARNING WARNING WARNING
WARNING WARNING WARNING WARNING WARNING WARNING WARNING WARNING WARNING WARNING WARNING WARNING
WARNING WARNING WARNING WARNING WARNING WARNING WARNING WARNING WARNING WARNING WARNING WARNING
WARNING WARNING WARNING WARNING WARNING WARNING WARNING WARNING WARNING WARNING WARNING WARNING
WARNING WARNING WARNING WARNING WARNING WARNING WARNING WARNING WARNING WARNING WARNING WARNING
WARNING WARNING WARNING WARNING WARNING WARNING WARNING WARNING WARNING WARNING WARNING WARNING
WARNING WARNING WARNING WARNING WARNING WARNING WARNING WARNING WARNING WARNING WARNING WARNING
WARNING WARNING WARNING WARNING WARNING WARNING WARNING WARNING WARNING WARNING WARNING WARNING
WARNING WARNING WARNING WARNING WARNING WARNING WARNING WARNING WARNING WARNING WARNING WARNING
WARNING WARNING WARNING WARNING WARNING WARNING WARNING WARNING WARNING WARNING WARNING WARNING
WARNING WARNING WARNING WARNING WARNING WARNING WARNING WARNING WARNING WARNING WARNING WARNING
WARNING WARNING WARNING WARNING WARNING WARNING WARNING WARNING WARNING WARNING WARNING WARNING
WARNING WARNING WARNING WARNING WARNING WARNING WARNING WARNING WARNING WARNING WARNING WARNING
WARNING WARNING WARNING WARNING WARNING WARNING WARNING WARNING WARNING WARNING WARNING WARNING
WARNING WARNING WARNING WARNING WARNING WARNING WARNING WARNING WARNING WARNING WARNING WARNING
WARNING WARNING WARNING WARNING WARNING WARNING WARNING WARNING WARNING WARNING WARNING WARNING
WARNING WARNING WARNING WARNING WARNING WARNING WARNING WARNING WARNING WARNING WARNING WARNING
```

This is due to the fact that the `dnssec-signzone` tool doesn't support the `-k` switch which would allow to make use of a key signing key (KSK) which is then forwarded to a parent zone to generate a DS record.

If you want to make use of this signed zone, change the filename in `named.conf` to "seafront.bar.signed"
# Mail and Lists

A wide variety of SMTP servers can run on Linux. The most popular servers are:

**Sendmail**
This server has long dominated Internet mail delivery. sendmail has also earned a reputation for a difficult-to-master configuration file format. Fortunately, tools to create a configuration file from a simpler file are common.

**Postfix**
This server is comparable to sendmail in popularity. Postfix uses a series of small programs to handle mail delivery tasks, as opposed to the monolithic approach used by sendmail. Its configuration is much easier to handle than is sendmail's.

**Exim**
This mail server is not quite as popular as sendmail or Postfix, but it is still a popular Linux mail server. Like sendmail, Exim uses a monolithic design, but Exim’s configuration file is much more intelligible. This server includes extensive pattern-matching tools that are very useful in fighting spam.

## 1. Using Sendmail

**Sendmail** is a general purpose internetwork email routing facility that supports many kinds of mail-transfer and delivery methods, including the **Simple Mail Transfer Protocol (SMTP)** used for email transport over the Internet.

### 1.1 Configuration Settings

**DNS Settings**
1. We first want to make sure that mail will be sent to our machine. We assume that we have properly configured a domain called seafront.bar with BIND 8 or 9. Let's make sure that the zone file for this domain has an MX record pointing to our system.

   For example if our machine is called **test1** and has the IP **192.168.246.12** then we need the following lines:

   ```
   seafront.bar. IN MX 10 test1.seafront.bar.
   test1.seafront.bar. IN A 192.168.246.12
   ```
2. Next we need to make sure that this information is read by the resolvers, so we add the following at the top of the file /etc/resolv.conf:

```bash	nameserver 127.0.0.1
domain seafront.bar
```

**Sendmail Settings**

We go into sendmail's main configuration directory /etc/mail. Here we need to do the following:

1. By default sendmail is configured to listen for connections ONLY for the 127.0.0.1 interface. In order to make sendmail listen to all interfaces we need to comment out the following line in /etc/mail/sendmail.mc using 'dnl' which stands for “do next line”:

   ```bash
dnl DAEMON_OPTIONS(`Port=smtp,Addr=127.0.0.1, Name=MTA')dnl
```

2. Once this is done run:

   ```bash
   m4 /etc/mail/sendmail.mc > /etc/mail/sendmail.cf
   ```

**Notice:** Make sure /etc/sendmail.cf isn't also there, if it is, delete it.

3. Restart sendmail and try the following:

   ```bash
telnet test1.seafront.bar 25
   ```

**Warning:** If you get a connection then sendmail is responding. This doesn't mean that sendmail will deliver mail (relay) for you!

3. To configure sendmail to relay for you you need to add the IP for your machine to the /etc/mail/access file:

   ```bash
   192.168.246.12 RELAY
   ```

4. Finally, we also need to tell sendmail to accept mail for @seafront.bar addresses. For this, add the domain name to /etc/mail/local-host-names:

   ```bash
   seafront.bar
   ```

Restart sendmail and send a mail to an existing user. If you have a user *tux* on the machine then check the output of the following:

```bash
mail -v -s "test seafront domain" tux@seafront.bar < /etc/passwd
```
1.2 Virtual Hosting

We want the server seafront.bar to accept mail for the city.bar domain. For this we follow the following steps.

The DNS entries

We need to add an MX record for the city.bar domain. Here is the whole block for clarity:

```
seafront.bar. IN MX 10 test1.seafront.bar.
city.bar. IN MX 10 test1.seafront.bar.
test1.seafront.bar. IN A 192.168.246.12
```

Reload the zone file:

```
rndc reload
```

Sendmail Settings

1. We need to make sendmail accept mail for users at @city.bar. For this we add the next line to the local-host-names file:

```
city.bar
```

If mail is sent to tux@city.bar and tux is a valid user on test1.seafront.bar then mail will be delivered to the local user tux.

To avoid this we can use the /etc/mail/virtusertable database.

2. If you want to forward mail onto another account here are example entries for the virtusertable database:

```
tux@city.bar   mr.tux@otherdomain.org
@city.bar      administrator
list@city.bar  local-list
```

Here mail for user tux is diverted to mr.tux@otherdomain.org, the user administrator is the catchall account and lists are redirected to local lists (this needs to point to a valid list defined in the aliases).
2. Configuring Mailing Lists

2.1 Majordomo and Sendmail

Pre-installation Configuration

1. In the Makefile, replace `/bin/perl` with the path to the perl binary on your system (usually `/usr/bin/perl`):

   ```bash
   PERL = /usr/bin/perl
   ```

   To make things easier we will leave the W_HOME as is:

   ```bash
   W_HOME = /usr/test/majordomo-$(VERSION)
   ```

   You need to create the directory `/usr/test`

   ```bash
   mkdir /usr/test
   ```

   Create a group called `majordomo` with GID 45, and add a user called `majordomo` with UID 123

   ```bash
   groupadd -g 45 majordomo
   useradd -g 45 -u 123 majordomo
   ```

2. In the `sample.cf` file we need to define our domain (for example `seafront.bar`). This is also where the path to the sendmail binary is set:

   ```bash
   $whereami = "seafront.bar";
   $sendmail_command = "/usr/sbin/sendmail";
   ```

   Now we can run

   ```bash
   make install
   make install-wrapper
   ```

   Finally you can test the configuration as suggested with the following:

   ```bash
   cd /usr/test/majordomo-1.94.5; ./wrapper config-test
   ```

   If all goes well you will be prompted to register to the majordomo mailing list. Since we do not have a valid email address, answer NO to the question.
Sendmail Configuration

The sendmail configuration involves adding appropriate entries in /etc/aliases for each mailing list we create. But before that we need a symbolic link in /etc/smrsh pointing to the majordomo wrapper binary, and here is why.

In order to limit the number of programs mail can be piped to (using a '|' command instead of an email address) sendmail defines a set of commands known as “sendmail restricted shells” or smrsh. The list of restricted shells is contained in /etc/smrsh which are symbolic links to the actual binaries we allow mail to be piped to.

We will make the wrapper binary available, which is located in /usr/test/majordomo-1.94.5, with the following:

```
ln -s /usr/test/majordomo-1.94.5/wrapper /etc/smrsh
```

Before adding the entries to /etc/aliases we need to decide on a name for our first list, and we choose ... test.

Remember that before sending mail to the list test@seafront.bar we first need to subscribe to this list by sending a mail to majordomo@seafront.bar with the contents subscribe test. Some work needs to be done for this to work.

Creating the list “test” (as documented in NEWLIST):

1. Create an empty file called test and a file containing information about the list called test.info in the directory /usr/test/majordomo-1.94.5/lists/

2. Create the following aliases in /etc/aliases:

   ```
   majordomo: "|/usr/test/majordomo-1.94.5/wrapper majordomo"
   test: "|/usr/test/majordomo-1.94.5/wrapper resend -l test test-list"
   test-list: :include:/usr/test/majordomo-1.94.5/lists/test
   test-request: "|/usr/test/majordomo-1.94.5/wrapper request-answer test"
   owner-test: tux
   test-approval: tux
   ```

3. Run newaliases and restart sendmail.

Majordomo Test

Send an email to majordomo@seafront.bar with the content:

```subscribe test```

If all goes well you will receive a response with further steps to be taken.
3. Managing Mail Traffic

3.1 Mail Filtering with Procmail

Procmail is a program for filtering electronic mail. It is very useful for presorting and preprocessing large amounts of incoming mail. You can use it to sort out mail from mailing lists, to dispose of junk mail, to send automatic replies, or even to run a mailing list.

The Procmail is generally not started from the command line. It is usually invoked by mail delivery subsystems (like Sendmail or Postfix) or from a mail retrieval agent (such as fetchmail). The companion tool formail allows Procmail to be used in batch-processing on mail that already is in a user's mailbox.

The primary use of Procmail is to filter messages into several mailboxes, based on the headers. This filtering is done based on the rules set down in your ~/.procmailrc file.

Procmail is normally installed on most distributions by default. Run which procmail to find out where Procmail is located (usually that is /usr/bin/procmail).

In depth information can be found in the procmail, procmailrc and procmailex manpages. Here are a few examples taken from procmailex(5).

The Procmail agent uses recipes, to determine where to deliver the various mail messages.

A promailrc file is a sequence of recipes of the form:

```
:0 [flags] [ : [locallockfile] ]
<zero or more conditions (one per line)>
<exactly one action line>
```

Each recipe that Procmail uses consists of flags, conditions and action. The next tables cover the main flags, conditions and actions available.

<table>
<thead>
<tr>
<th>Flags</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>H</td>
<td>Egrep the header (default).</td>
</tr>
<tr>
<td>B</td>
<td>Egrep the body</td>
</tr>
<tr>
<td>E</td>
<td>This recipe only executes if the immediately preceding recipe was not executed.</td>
</tr>
<tr>
<td>e</td>
<td>This recipe only executes if the immediately preceding recipe failed</td>
</tr>
<tr>
<td>w</td>
<td>Wait for the filter or program to finish and check its exit code</td>
</tr>
</tbody>
</table>
The conditions are extended regular expressions with the additional conditions below:

<table>
<thead>
<tr>
<th>Conditions</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>!</td>
<td>Invert the condition</td>
</tr>
<tr>
<td>$</td>
<td>Evaluate the remainder of this condition according to sh(1) substitution rules inside double quotes, skip leading whitespace, then reparse it</td>
</tr>
<tr>
<td>?</td>
<td>Use the exitcode of the specified program</td>
</tr>
<tr>
<td>&lt;</td>
<td>Check if the total length of the mail is shorter than the specified (in decimal) number of bytes</td>
</tr>
<tr>
<td>&gt;</td>
<td>Check if the total length of the mail is larger than the specified (in decimal) number of bytes</td>
</tr>
</tbody>
</table>

The action line can start with one of

<table>
<thead>
<tr>
<th>Action line</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>!</td>
<td>Forwards to all the specified mail addresses</td>
</tr>
<tr>
<td></td>
<td></td>
</tr>
<tr>
<td>{</td>
<td>Followed by at least one space, tab or newline will mark the start of a nesting block</td>
</tr>
<tr>
<td>Anything else</td>
<td>interpret as a mailbox (file or directory relative to current directory or MAILDIR)</td>
</tr>
</tbody>
</table>

Examples:

Sort all mail coming from the lpi-dev mailing list into the mail folder LPI:

```
:0:
* ^To.*lpi-dev
LPI
```

Forward mails between two accounts main.address and the-other.address. This rule is for the procmailrc on the main address account. Notice the X-Loop header used to prevent loops:

```
:0 c
* !^X-Loop: yourname@main.address
| formail -A "X-Loop: yourname@main.address" | \ $SENDMAIL -oi yourname@the-other.address
```

The c option tells Procmail to keep a local copy.
Filtering Spam

Spam filtering generally takes place using SpamAssassin. This program analyzes your mail for its likely level of "spamminess", and gives it a numeric score; anything greater than 5 is generally considered spam, and is marked up as such, by adding an "X-Spam-Status: yes" header.

The new mail server automatically runs all emails through SpamAssassin. You can filter messages that SpamAssassin marks as spam by adding the following recipe to your .procmailrc file:

```
:0:
* ^X-Spam-Status: yes
spam
```
Web Services

1. Implementing a Web Server

1.1 Installing Apache

The apache source code can be downloaded from www.apache.org.

There are two versions of the apache server: 1.3 and 2.x

The configure script allows us to customise the installation. In particular we can choose which modules we want to compile etc. Modules can either be

- statically compiled with
  --enable-MODULE (where MODULE is the Module Identifier) or
  --enable-modules="MOD1 MOD2 ..."

- dynamically compiled with
  --enable-mods-shared="MOD1 MOD2 ..."

- disabled with
  --disable-MODULE

1.2 Monitoring apache load

SNMP

Create a read-only SNMP community and restart the snmpd daemon:

<table>
<thead>
<tr>
<th>/etc/snmp/snmp.conf</th>
</tr>
</thead>
<tbody>
<tr>
<td>rocommunity lifesavers</td>
</tr>
</tbody>
</table>

Restart the snmpd service:

| /etc/init.d/snmpd restart |

Check that you can browse information about your system using the community name lifesavers:

| snmpwalk -v 1 -c lifesavers localhost ip |
MRTG

MRTG stands for “multi-router traffic grapher” and uses SNMP to get information about the system.

```bash
|cfgmaker --output=/etc/mrtg/seafront.cfg \|
   -ifref=ip --global "workdir: /var/www/mrtg/stats" \|
   lifesavers@localhost
```

This will create a file called /etc/mrtg/seafront.cfg. We next update the information in /var/www/mrtg/stats with the following command:

```bash
mkdir /var/www/mrtg/stats
mrtg /etc/mrtg/seafront.cfg
```

This should be run at regular intervals so it should be run through a cron job.

**Task:** The graphical output for MRTG will be saved in /var/www/mrtg/stats as an HTML document. This is not a usual place to keep files for the apache server. After the next section, we will make the appropriate changes to `httpd.conf` to make this directory accessible through the webserver.

Many other tools are available such as **Webalizer** which analyse the access logs of the apache server.

### 1.3 Using Apachectl

The **apachectl** script is used to control the **httpd** daemon. It takes the following options:

<table>
<thead>
<tr>
<th>apachectl option</th>
<th>Description – extract from apachectl(8)</th>
</tr>
</thead>
<tbody>
<tr>
<td>start</td>
<td>Start the Apache httpd daemon. Gives an error if it is already running. This is equivalent to <code>apachectl -k start</code></td>
</tr>
<tr>
<td>stop</td>
<td>Stops the Apache httpd daemon. This is equivalent to <code>apachectl -k stop</code></td>
</tr>
<tr>
<td>restart</td>
<td>Restarts the Apache httpd daemon. If the daemon is not running, it is started. This command automatically checks the configuration files as in configtest before initiating the restart to make sure the daemon doesn’t die. This is equivalent to <code>apachectl -k restart</code></td>
</tr>
</tbody>
</table>
fullstatus | Displays a full status report from mod_status. For this to work, you need to have mod_status enabled on your server and a text-based browser such as lynx available on your system. The URL used to access the status report can be set by editing the STATUSURL variable in the script.

status | Displays a brief status report. Similar to the fullstatus option, except that the list of requests currently being served is omitted.

graceful | Gracefully restarts the Apache httpd daemon. If the daemon is not running, it is started. This differs from a normal restart in that currently open connections are not aborted. This is equivalent to `apachectl -k graceful`.

configtest | Run a configuration file syntax test. It parses the configuration files and either reports Syntax Ok or detailed information about the particular syntax error. This is equivalent to `apachectl -t`.

### 1.4 Basic Configuration Options

#### Section 1: General Options

<table>
<thead>
<tr>
<th>Option</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>KeepAlive on/off</td>
<td>Allows a client to perform multiple requests through a single connection</td>
</tr>
<tr>
<td>MaxKeepAliveRequests 100</td>
<td>Maximum number of requests during a persistent connection</td>
</tr>
<tr>
<td>KeepAliveTimeout 15</td>
<td>Number of seconds to wait for a next request on the same connection</td>
</tr>
</tbody>
</table>

#### Single Threaded Server

The httpd daemon is a single threaded process which needs to fork child daemons to deal with multiple connections – only with apache2 is it possible to build a multi threaded server.

<table>
<thead>
<tr>
<th>Option</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>StartServers 8</td>
<td>Number of httpd servers to start</td>
</tr>
<tr>
<td>MinSpareServers 5</td>
<td>Minimum number of spare servers to keep loaded in memory</td>
</tr>
<tr>
<td>MaxSpareServers 20</td>
<td>Maximum number of spare servers to keep loaded in memory</td>
</tr>
<tr>
<td>MaxClients 150</td>
<td>Maximum number of server processes allowed at any one time</td>
</tr>
<tr>
<td>MaxRequestsPerChild 1000</td>
<td>Maximum number of requests before a child is “retired”</td>
</tr>
</tbody>
</table>
Multi Threaded Server

These options are available only for apache2 and onwards. You need to recompile apache to enable threads. Most current apache2 binary distributions are still single threaded because of conflicts with most dynamic modules which don’t support multi threading yet.

<table>
<thead>
<tr>
<th>Option</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>StartServers 2</td>
<td>Notice that this is much lower than the single threaded server</td>
</tr>
<tr>
<td>MinSpareThreads 25</td>
<td>Minimum number of spare threads</td>
</tr>
<tr>
<td>MaxSpareThreads 75</td>
<td>Maximum number of spare threads</td>
</tr>
<tr>
<td>ThreadsPerChild 25</td>
<td>Number of worker threads per child</td>
</tr>
<tr>
<td>MaxClients 150</td>
<td>Maximum number of server processes allowed at any one time</td>
</tr>
<tr>
<td>MaxRequestsPerChild 0</td>
<td>Never retires?</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Option</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Listen 80</td>
<td>Specify which port to listen on. Can be of the form IP:port</td>
</tr>
<tr>
<td>LoadModule MODULE IDENTIFIER /PATH-TO/MODULE</td>
<td>Section where dynamic modules are loaded</td>
</tr>
<tr>
<td>Include FILE</td>
<td>Read extra configuration options from FILE. Apache2 has a conf.d directory for this</td>
</tr>
</tbody>
</table>

**Section 2: Server Configuration**

- **ServerName**
  - The name of the server – can be different
- **User**
  - Name of the user the server runs as
- **Group**
  - Name of the group the server runs as
- **DocumentRoot**
  - The directory the where HTML files are kept
- **<Directory>**
  - Specify options (access control,...) for directories containing HTML files
- **Alias**
  - URL alias for a given directory
- **AliasScript**
  - Same as “Alias” option but for directories containing CGI scripts
- **DirectoryIndex**
  - Set the name of the file which will be used as an index

**Section 3: Virtual Hosts**

We will cover virtual hosts when configuring SSL servers later in this chapter. For now we distinguish two concepts:

- **<VirtualHost IP:PORT>**
  - IP based virtual host
- **<VirtualHost HOSTNAME:PORT>**
  - Name based virtual
1.5 Restricting Client Access

Host based control is available using the keywords Order, Deny from and Allow from on directories

```html
<Directory PATH-TO-DIRECTORY> ... </Directory>
```
or locations

```html
<Location URL> ... </Location>
```

The next configuration paragraph will allow anybody to access the directory /var/www/safe except the host with IP 192.168.3.101:

```html
<Directory /var/www/safe>
Order allow,deny
Deny from 192.168.3.101
Allow from all
</Directory>
Alias /safe /var/www/safe
```

Notice: The Order keyword is important. If we reverse the above order to Order deny,allow then the following would happen: host 192.168.3.101 would first be denied access because of the Deny rule but the Allow rule is read last and will subsequently grant it access. The default access is given by the last argument in the order directive. I.e. “Order allow,deny” has a default of “deny”.

1.6 Client Basic Authentication

The htpasswd tool is used to create passwords for users. For example, we create a new file in the ServerRoot directory called passwords-for-directory1 with a password for user gnu:

```bash
htpasswd -c passwords-for-directory1 gnu
```

If we choose to implement client authentication for the directory /var/www/html/seafront we need to add the following paragraph to httpd.conf:
<Directory /var/www/html/seafront>
AuthType basic
AuthName "protected site"
AuthUserFile conf/seafront.passwd
Require user gnu
</Directory>

Notice: Alternatively, with httpd2 configurations we could create a file called seafront.conf with the above content and save it in the /etc/httpd/conf.d directory.

Reread the configuration file with:

```
apachectl graceful
```

## 2. Maintaining a Web Server

### 2.1 HTTPS Overview

The secure socket layer protocol SSL allows any networked applications to use encryption. This can be thought of as a process which wraps the socket preparing it to use encryption at the application level.

In the case of HTTPS, the server uses a pair of keys, public and private. The server's public key is used by the client to encrypt the session key, the private key is then used to decrypt the session key for use.

The public key is published using certificates. A certificate contains the following information:

- Name and Address, Hostname, etc.
- Public Key
- TTL
- (optional) ID + Signature from a certificate authority (CA)

The certificate will be used to establish the authenticity of the server. A valid signature from a known CA is automatically recognised by the client's browser. With Mozilla for example these trusted CA certificates can be found by following the links: Edit -> Preferences -> Privacy & Security -> Certificates then clicking on the “Manage Certificates” button and the Authorities tab.
On the other hand communications would be too slow if the session was encrypted using public key encryption. Instead, once the authenticity of the server is established, the client generates a unique secret session key which is encrypted using the servers public key found in the certificate. Once the server receives this session key it can decrypt it using the private key associated with the certificate. From there on the communication is encrypted and decrypted using this secret session key generated by the client.

2.2 SSL Virtual Hosts

A separate apache server can be used to listen on port 443 and implement SSL connections. However most default configurations involve a single apache server listening on both ports 80 and 443.

For this an additional **Listen** directive is set in **httpd.conf** asking the server to listen on port 443. Apache will then bind to both ports 443 and 80. Non encrypted connections are handled on port 80 while an SSL aware virtual host is configured to listen on port 443:

```
<VirtualHost _default_:443>

SSL CONFIGURATION

</VirtualHost>
```
The SSL CONFIGURATION lines are:

```
SSLEngine on
SSLCertificateFile PATH_TO_FILE.crt
SSLCertificateKeyFile PATH_TO_FILE.key
```

We need to generate the servers private key (FILE.key) and certificate (FILE.crt) to complete this configuration.

### 2.3 Managing Certificates

The keys and certificates are usually kept in subdirectories of `/etc/httpd/conf` called `ssl.crt` and `ssl.key`.

There should also be a Makefile that will generate both a KEY and a CERTIFICATE in PEM format which is base64 encoded data.

#### Using the Makefile

For example if we want to generate a self-signed certificate and private key simply type:

```
make mysite.crt
```

The Makefile will generate both files mysite.key (the private key) as well as mysite.crt (the certificate file containing the public key). You can use the following directives in `httpd.conf`:

```
SSLCertificateFile ... mysite.crt
SSLCertificateKeyFile ... mysite.key
```

#### Certificate Requests

On a production server you would need to generate a new file called a “certificate request” with:

```
openssl req -new -key mysite.key -out mysite.csr
```

This file can be sent to a certificate authority (CA) to be signed. The certificate authority will send back the signed certificate.
Pass Phrases

A private key can be generated with or without a passphrase, and a private key without a passphrase can be constructed from an existing private key.

A passphrase file: If a private key has a passphrase set then the file starts with

```
-----BEGIN RSA PRIVATE KEY-----
Proc-Type: 4,ENCRYPTED
DEK-Info: DES-EDE3-CBC, ---- snip ----
```

this means that the file is protected by a pass-phase using 3DES. This was generated by the line `/usr/bin/openssl genrsa -des3 1024 > $@` in the Makefile. If the `-des3` flag is omitted NO passphrase is set.

You can generate a new private key (mysite-nophrase.key) without a passphrase from the old private key (mysite.key) as follows:

```
openssl rsa -in mysite.key -out mysite-nopass.key
```

2.4 Virtual Hosts

Name based virtual hosts

We will first discuss the situation where only one IP has been assigned to the server but there are several A records or CNAME records pointing to the same IP.

Example: Modify the zone files to include a new CNAME record for `test1.seafront.bar` to point to the actual name of the web server.

```
e.g. test1.seafront.bar. IN CNAME server1.seafront.bar.
     server1 IN A 192.x.x.x
```

In httpd.conf it will be enough to create the following:

```
<VirtualHost test1.seafront.bar:80>
   ServerAdmin webmaster@seafront.bar
   DocumentRoot /var/www/html/test1
   ServerName test1.seafront.bar
</VirtualHost>
```
**Example 2: Create an SSL aware VirtualHost for test1**

- **Make the certificate and the key:** make host1.seafront.bar
- **Add these lines to httpd.conf:**

  ```
  <VirtualHost 192.168.3.200:443>
  SSLEngine on
  SSLCertificateFile /etc/httpd/conf/test1.seafront.bar.crt
  SSLCertificateKeyFile /etc/httpd/conf/test1.seafront.bar.out
  ServerAdmin webmaster@seafront.bar
  DocumentRoot /var/www/html/test1
  ServerName test1.seafront.bar
  </VirtualHost>
  ```

Notice that the certificate that is presented once you connect to the https://test1 site is incorrect. This is because test1.seafront.bar resolves to the servers IP address and the server will start the SSL handshake before looking at the HTTP request. The next section will fix that.

**IP Based Virtual Hosts**

**Example:** We will directly create a series of virtual SSL aware hosts and verify that they present the client with the correct certificate.

- **Assign new IP addresses to the eth0 interface:** ifconfig eth0:0 X.X.X.X
- **For each IP enter a new A record:** www1 IN A X.X.X.X
- **For each host create a self signed certificate.**
- **Enter a `<VirtualHost X.X.X.X:443>` paragraph in httpd.conf.**

**Notice:** You may have to change the existing SSL virtual host from

```
<VirtualHost _default_:443>
  to
  <VirtualHost 127.0.0.1:443>
```

This prevents the default host certificate from being presented irrespective of the site hostname.

Test that https://www1 and https://www2 do present the proper certificates. Notice that if you permanently accept a certificate it will be added to the list of CA certificates on your browser!
3. Implementing a Proxy Server

3.1 Getting Started

You can verify that the squid proxy server is installed using:

```
rpm -q squid
```

Most versions will install the `/etc/init.d/squid` rc-script that creates the initial caching directories. If this is not the case squid can initialise these cache directories with the `-z` switch.

```
squid -z
```

**NOTICE**

You may need to add an access rule in the squid configuration file before being able to rebuild the cache (see the next section “Access Lists and Access Control”).

The configuration file is `/etc/squid/squid.conf`. The syntax of this file can be checked using the `-k` switch:

```
squid -k check
```

As with most network services the `/etc/init.d/squid` rc-script is used to start the service.

3.2 Access Lists and Access Control

- **Access Lists (acl)**

In `squid.conf` the access lists have the following format:

```
  acl aclname acltype string/file
```

In the simplest cases an `acl` defines a list of hosts, networks or domains and is given a name. This list can then be granted or denied access using the access control command `http_access` described in the next paragraph.
The next line defines an access list name called *localnet* corresponding to the local LAN:

```
ac1 localnet src 192.168.2.0/255.255.255.0
```

The main ACL types are listed below:

<table>
<thead>
<tr>
<th>acltype</th>
<th>description</th>
</tr>
</thead>
<tbody>
<tr>
<td>src</td>
<td>IP/netmask or IP1-IP2/netmask (client's IP address)</td>
</tr>
<tr>
<td>dst</td>
<td>IP/network (URL requested)</td>
</tr>
<tr>
<td>arp</td>
<td>MAC address</td>
</tr>
<tr>
<td>srcdomain</td>
<td>.example.com (client addresses)</td>
</tr>
<tr>
<td>dstdomain</td>
<td>.example.com (URLs requested)</td>
</tr>
<tr>
<td>time</td>
<td>range of times</td>
</tr>
<tr>
<td>port</td>
<td>space separated list of ports or range of the form p1-p2</td>
</tr>
</tbody>
</table>

- **Access control (http_access)**

With *http_access* a particular access list is either allowed or denied access via the proxy. The format is as follows:

```
http_access allow|deny aclname
```

The http_access requests are read in sequence and the first rule matched is used. To allow access to all computers on the network insert the following *before* the `http_access deny all` line:

```
http_access allow localnet
```
3.3 Additional Configuration Options

The following table is a list of additional options available to further control the squid proxy.

<table>
<thead>
<tr>
<th>Option</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>http_port</td>
<td>the port squid uses to listen for requests (default 3128)</td>
</tr>
<tr>
<td>cache_peer</td>
<td>specify another proxy server to query whenever an object isn't cached</td>
</tr>
<tr>
<td>cache_mem</td>
<td>limit the amount of additional memory used to cache objects (this parameter doesn't limit the maximum process size)</td>
</tr>
<tr>
<td>cache_swap_low</td>
<td>percentage of swap utilisation. Once this limit is passed objects start to be cached to disk</td>
</tr>
<tr>
<td>cache_swap_high</td>
<td>percentage of swap utilisation. Once this limit is approached objects start getting evicted from the proxy cache</td>
</tr>
<tr>
<td>maximum_object_size</td>
<td>objects larger than this will not be cached</td>
</tr>
<tr>
<td>maximum_object_size_in_memory</td>
<td>objects larger than this will not be kept in the memory cache</td>
</tr>
</tbody>
</table>

Memory Management (from the SQUID FAQ section 8)

“This version of SQUID stores incoming objects only in memory, until the transfer is complete. At that point it decides whether or not to store the object on disk. This means that when users download large files, your memory usage will increase significantly. The squid.conf parameter maximum_object_size determines how much memory an in-transit object can consume before we mark it as uncachable. When an object is marked uncachable, there is no need to keep all of the object in memory, so the memory is freed for the part of the object which has already been written to the client. In other words, lowering maximum_object_size also lowers Squid-1.1 memory usage.”

“If your cache performance is suffering because of memory limitations, you might consider buying more memory. But if that is not an option, There are a number of things to try:

- Try a different malloc library  [compile SQID with a different malloc]
- Reduce the cache_mem parameter in the config file. This controls how many "hot" objects are kept in memory. Reducing this parameter will not significantly affect performance, but you may receive some warnings in cache.log if your cache is busy
- Turn the memory_pools off in the config file. This causes Squid to give up unused memory by calling free() instead of holding on to the chunk for potential, future use.
- Reduce the cache_swap parameter in your config file. This will reduce the number of objects Squid keeps. Your overall hit ratio may go down a little, but your cache will perform significantly better
- Reduce the maximum_object_size parameter (Squid-1.1 only). You won’t be able to cache the larger objects, and your byte volume hit ratio may go down, but Squid will perform better overall"
3.4 Reporting Tools

Most log analysis tools available for squid are listed on the following site:

http://www.squid-cache.org/Scripts/

The main log file for squid is the /var/log/squid/access.log file. Next is a short overview of calamaris and webalizer. Also notice that webmin produces log reports based on calamaris.

**cachemgr.cgi script**

The current squid package installs a CGI script in /usr/lib/squid called cachemgr.cgi. One can copy this across to the /var/www/cgi-bin directory where all CGI scripts can run from. It is recommended however to set up a separate directory with htaccess authentication.

**Calamaris**

The code is GPL and can be downloaded from http://cord.de/tools/squid/calamaris. You can generate reports as follow:

```bash
cat /var/log/squid/access.log | calamaris
```

```plaintext
# Summary
lines parsed: 221
invalid lines: 0
parse time (sec): 0

# Incoming requests by method
method request % Byte % sec kB/sec
----------------------------------------
GET 221 100.00 1244262 100.00 3 1.68
----------------------------------------
Sum 221 100.00 1244262 100.00 3 1.68

# Incoming UDP-requests by status
no matching requests

# Incoming TCP-requests by status
status request % Byte % sec kB/sec
-------------------------------
HIT 35 15.84 42314 3.40 0 6.11
MISS 182 82.35 1197840 96.27 1 4.97
ERROR 4 1.81 4108 0.33 120 0.01
-------------------------------
Sum 221 100.00 1244262 100.00 3 1.68
```

In order to get information on webpage requests per host one can use the -R switch: There are many more switches available (check the manpages for calamaris).
There are also a number of scripts that can run hourly or monthly reports. These scripts are included in the EXAMPLES file distributed with calamaris.

calamaris -R 5 /var/log/squid/access.log

<table>
<thead>
<tr>
<th>host / target</th>
<th>request</th>
<th>hit-%</th>
<th>Byte</th>
<th>hit-% sec</th>
<th>kB/sec</th>
</tr>
</thead>
<tbody>
<tr>
<td>192.168.2.103</td>
<td>72</td>
<td>0.00</td>
<td>323336</td>
<td>0.00</td>
<td>0</td>
</tr>
<tr>
<td>*.redhat.com</td>
<td>35</td>
<td>0.00</td>
<td>126726</td>
<td>0.00</td>
<td>0</td>
</tr>
<tr>
<td>*.suse.co.</td>
<td>20</td>
<td>0.00</td>
<td>63503</td>
<td>0.00</td>
<td>0</td>
</tr>
<tr>
<td>*.lemonde.fr</td>
<td>6</td>
<td>0.00</td>
<td>109712</td>
<td>0.00</td>
<td>1</td>
</tr>
<tr>
<td>207.36.15.*</td>
<td>5</td>
<td>0.00</td>
<td>8946</td>
<td>0.00</td>
<td>0</td>
</tr>
<tr>
<td>*.akamai.net</td>
<td>4</td>
<td>0.00</td>
<td>12428</td>
<td>0.00</td>
<td>1</td>
</tr>
<tr>
<td>other: 2 requested urlhosts</td>
<td>2</td>
<td>0.00</td>
<td>2021</td>
<td>0.00</td>
<td>1</td>
</tr>
<tr>
<td>192.168.2.101</td>
<td>63</td>
<td>0.00</td>
<td>295315</td>
<td>0.00</td>
<td>1</td>
</tr>
<tr>
<td>cord.de</td>
<td>17</td>
<td>0.00</td>
<td>115787</td>
<td>0.00</td>
<td>0</td>
</tr>
<tr>
<td>*.doubleclick.net</td>
<td>13</td>
<td>0.00</td>
<td>26163</td>
<td>0.00</td>
<td>1</td>
</tr>
<tr>
<td>*.google.com</td>
<td>10</td>
<td>0.00</td>
<td>30646</td>
<td>0.00</td>
<td>1</td>
</tr>
<tr>
<td>*.squid-cache.org</td>
<td>8</td>
<td>0.00</td>
<td>51758</td>
<td>0.00</td>
<td>1</td>
</tr>
<tr>
<td>&lt;error&gt;</td>
<td>4</td>
<td>0.00</td>
<td>4290</td>
<td>0.00</td>
<td>0</td>
</tr>
<tr>
<td>other: 6 requested urlhosts</td>
<td>11</td>
<td>0.00</td>
<td>66671</td>
<td>0.00</td>
<td>5</td>
</tr>
</tbody>
</table>

Sum: 135, 0.00, 618651, 0.00, 1, 6.51

Webalizer

This tool is often installed by default on some Linux distributions. It is also GPL'ed and can be downloaded from http://www.mrunix.net/webalizer/.

By editing the /etc/webalizer.conf file one can choose between apache access logs, ftp transfer logs or squid logs.

Example graphics generated with Webaliser:
3.5 User Authentication (using PAM)

To prevent unauthorised users browsing on the Internet you can setup squid to ask for a username and password.

IMPORTANT: You cannot have user authentication and transparent proxy at the same time! The work around is to block all outgoing requests on port 80, except the ones from the Squid proxy itself. Users are then forced to manually set up their browsers to use the proxy.

Configuration settings for PAM authentication:

Here are the list of options you need to set in the `squid.conf` file:

<table>
<thead>
<tr>
<th>squid.conf</th>
<th>PAM authentication settings</th>
</tr>
</thead>
<tbody>
<tr>
<td>[Older versions]</td>
<td>authenticate_program /usr/lib/squid/pam_auth</td>
</tr>
<tr>
<td>[Squid V2.5]</td>
<td>auth_param basic program /usr/lib/squid/pam_auth</td>
</tr>
<tr>
<td></td>
<td>auth_param basic children 5</td>
</tr>
<tr>
<td></td>
<td>auth_param basic realm Anvil Internet Proxy</td>
</tr>
<tr>
<td></td>
<td>auth_param basic credentialsttl 2 hours</td>
</tr>
<tr>
<td>acl password proxy_auth REQUIRED</td>
<td></td>
</tr>
<tr>
<td>http_access allow password</td>
<td></td>
</tr>
</tbody>
</table>

The PAM configuration in `/etc/pam.d`:

Here we register squid to use the Pluggable Authentication Module. This is done by adding a file in `/etc/pam.d/` called `squid` with the following content:

```
/etc/pam.d/squid

auth required /lib/security/pam_stack.so service=system-auth
auth required /lib/security/pam_nologin.so
account required /lib/security/pam_stack.so service=system-auth
password required /lib/security/pam_stack.so service=system-auth
session required /lib/security/pam_stack.so service=system-auth
session required /lib/security/pam_limits.so
```

This is a standard policy description on what to do when a person logs on. The login session is abstracted into 4 part: auth, account, password and session.

PAM then uses a specific library function which handles each stage. Notice that most lines request the `system-auth` service which is the
/etc/pam.d/system-auth file.

Also note the following from the pam_auth man page.

When used for authenticating to local UNIX shadow password databases the program must be running as root or else it won’t have sufficient permissions to access the user password database. Such use of this program is not recommended, but if you absolutely need to then make the program setuid root

```
chown root pam_auth
chmod u+s pam_auth
```

Please note that in such configurations it is also strongly recommended that the program is moved into a directory where normal users cannot access it, as this mode of operation will allow any local user to brute-force other users passwords. Also note the program has not been fully audited and the author cannot be held responsible for any security issues due to such installations.
Network Client Management

1. DHCP Configuration

Dynamic Host Configuration Protocol (DHCP) is network protocol for automatically assigning TCP/IP information to client machines. Each DHCP client connects to the centrally-located DHCP server that returns the client's network configuration including IP address, gateway, and DNS servers.

**WARNING:** You should not attempt to run a DHCP server unless you are certain not to interfere with the network you are currently using – The safest option for this section is to be totally isolated from the network and use a hub or a switch to connect the classroom together.

1.1 Default DHCP Configurations

The basic communication process between a client workstation joining a TCP/IP network and the DHCP server is depicted below.

The DHCPDISCOVER request is sent using the broadcast 255.255.255.255

The DHCP server can use two methods to allocate IP addresses:

1. A dynamic IP is assigned for a client host chosen from a range of IPs
2. A fixed IP is assigned for a specific host (identified using the MAC address, similar to bootp)
Since a single DHCP server can be used to administer IPs over several networks, the `dhcpd.conf` configuration file is composed of global options followed by network sections:

Example network block:

```bash
subnet 10.0.0.0 netmask 255.0.0.0 {
    ....
}
```

In the next example we will assign both dynamic IP addresses and a fixed IP address:

```bash
subnet 10.0.0.0 netmask 255.0.0.0 {
    range 10.5.5.10 10.5.5.200;
    host proxy {
        hardware ethernet 00:80:C6:30:0A:7E;
        fixed-address 10.5.5.2;
    }
}
```

For each subnet it is possible to give information on network services, such as:

- the default gateway
- the DNS domain name and the NIS domain name
- the DNS servers

In the subnet section above these directives would look like this:

```bash
    option routers                  10.254.254.254;
    option nis-domain               "nisdomain";
    option domain-name               "seafront.bar";
    option domain-name-servers       10.0.0.2;
```

The database of dynamically assigned IP addresses is stored in `/var/lib/dhcp/dhcpd.leases`.

This file should not be modified by hand. DHCP lease information for each recently assigned IP address is automatically stored in the lease database. The information includes the length of the lease, to whom the IP address has been assigned, the start and end dates for the lease, and the MAC address of the network interface card that was used to retrieve the lease.
NOTICE

When the DHCP server is started for the first time, it fails unless the dhcpd.leases file exists. Use the command touch /var/lib/dhcpd/dhcpd.leases to create the file if it does not exist.

If the same server is also running BIND as a DNS server, this step is not necessary, as starting the named service automatically checks for a dhcpd.leases file.

To start the DHCP service, use the command /sbin/service dhcpd start. To stop the DHCP server, use the command /sbin/service dhcpd stop.

If more than one network interface is attached to the system, but the DHCP server should only be started on one of the interfaces, configure the DHCP server to start only on that device. In /etc/sysconfig/dhcpd, add the name of the interface to the list of DHCPDARGS:

```
DHCPDARGS=eth0
```

### 1.2 Dynamic DNS

We assume that we still have the private/public key used for the seafront TSIG authentication, we will use this same key to allow the DHCP server to update the zone files on the DNS server.

**Additional Configurations on the DHCP Server**

On the DHCP server add the following to the dhcpd.conf file

```diffns-update-style interim;
ignore client-updates;
key seafront.bar. {
    algorithm hmac-md5;
    secret QN3vlApnV76WS+a2Hr3qj+AqZjpuPjQgVWeeMMGSBC4=;
};

zone seafront.bar. {
    primary 192.168.3.100;
    key seafront.bar.;
}

zone 3.168.192.in-addr.arpa. {
    primary 192.168.3.100;
    key seafront.bar.;
}
```
Optionally, it is possible to set a specific host name and domain name for a given host with the keywords

```
ddns-hostname host_name
ddns-domain-name domain_name
```

If the `ddns-hostname` option is not present then the DHCP server will try and use the name provided by the client. The domain on the other hand cannot be set by the client, so if `ddns-domain-name` is not present then the DHCP server will use the value given by the `domain-name` option.

**Additional Configurations on the DNS Server**

On the DNS server we need to do the following:

1. If you are using DNSSEC signed zone files then we need to use the unsigned zones.
2. Add the `allow-update` option to the `seafront.bar` entry:

   ```
   zone "seafront.bar" IN {
       type master;
       file "seafront.zone";
       allow-update { key seafront.bar.; };
       allow-transfer { key seafront.bar.; };
   }
   ```

   and do the same with the `in-addr.arpa` zone:

   ```
   zone "3.168.192.in-addr.arpa" IN {
       type master;
       file "192.168.3.local";
       allow-update { key seafront.bar.; };
       allow-transfer { key seafront.bar.; };
   }
   ```

**Testing with nsupdate**

`nsupdate` is a utility used to request the name server to update its database. `nsupdate` takes commands like `nslookup` does, if run without arguments.
The following commands are good to know:

- **server [server address]** - Sets the target server for who to send updates.
- **key [keyname] [secret]** - Tell nsupdate what your key is.
- **zone [zonename]** - Explicitly choose a zone to send updates for. If unspecified, nsupdate will guess.
- **update [...]** - Request an update to record.
- **send** - Send updates.
- **show** - Show updates that haven't been sent.

### Client Configuration

On Linux clients it is possible to set the DHCP_HOSTNAME variable in the interface setup script. In Redhat-like variants this would be in the `/etc/sysconfig/network-scripts/ifcfg-ethX` files. Notice that this is simple a hostname, the domain name will be appended to that name on the DHCP server.

### 1.3 DHCP Relay

The DHCPDISCOVER packets from clients reach the server through the broadcast 255.255.255.255, however broadcasts are blocked by routers.

So in a configuration with multiple networks and a single DHCP server each router needs to be able to relay DHCPDISCOVER broadcasts from a given network to the DHCP server.

For a Linux router this is done using the `dhcprelay` or `dhcrelay` (more recent) tool. Both tools take a mandatory single argument which is the IP of the DHCP server.

By default the relay tools will listen on all network interfaces for DHCP requests. One can specify an interface with the `-i` option:

```
dhcrelay -i eth0 IP_FOR_DHCP_server
```
2. NIS Configuration

2.1 Master Server Configuration

On a Linux system the network information system (NIS) server is called ypserv (package name: ypserv). The RPM package has the same name and installs the following main files:

<table>
<thead>
<tr>
<th>Files installed with ypserv</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>/etc/rc.d/init.d/yppasswdd</td>
<td>script for the daemon allowing users to change passwords</td>
</tr>
<tr>
<td>/etc/rc.d/init.d/ypserv</td>
<td>script for ypserv daemon</td>
</tr>
<tr>
<td>/etc/rc.d/init.d/ypxfrd</td>
<td>script for daemon used to speed up transfers to slave servers</td>
</tr>
<tr>
<td>/etc/ypserv.conf</td>
<td>main configuration file for ypserv</td>
</tr>
<tr>
<td>/var/yp/Makefile</td>
<td>Makefile for database files – should only be used on the master server</td>
</tr>
</tbody>
</table>

1. Choose a nisdomain name

   In /etc/sysconfig/network set the variable NISDOMAIN. For example we can set the nisdomain to linis as follows:

   NISDOMAIN=linis  # entry in /etc/sysconfig/network

   The file /etc/sysconfig/network will be sourced by the ypserv initscript.

2. Make sure the master server will push map changes to the slave servers. For this you need to edit the file /var/yp/Makefile and put

   NOPUSH=false

3. Start the ypserv daemon

   /etc/init.d/ypserv restart

4. Check that the nisdomain has been properly set

   nisdomainname
   linis
5. Create the databases, the -m option to `ypinit` is to indicate the server is a master server

```
/usr/lib/yp/ypinit -m
```

Enter the list of slave servers you will run on this domain. This will create a number of DBM files in `var/yp/linis` as well as a file called `var/yp/ypservers`.

### 2.2 Slave Server Configuration

On the slave server, we need to install the `ypserv` package too. This time we run `ypinit` and point it to the master server:

```
/etc/rc.d/init.d/ypserv start
/usr/lib/yp/ypinit -s MASTER_IP
```

Also make sure to leave the line `NOPUSH=true` in `var/yp/Makefile`.

### 2.3 Client Setup

On the client the main service is called `ypbind` (package name: ypbind). This daemon is responsible for binding to a NIS server and successfully resolves names and passwords as needed.

The main configuration file is `/etc/yp.conf`.

If the NISDOMAIN variable is set in `/etc/sysconfig/network` which is sourced by the `rc`-script `/etc/init.d/ypbind` then the NIS server will be detected using the broadcast. One can also configure `yp.conf` and specify. Once this is set one can start `ypbind`.

```
/etc/init.d/ypbind start
```

Make sure that the `nis` keyword is added to `/etc/nsswitch.conf`. 
2.4 Setting up NFS home directories

Once the NIS server and clients are setup as above, anybody with an account on the NIS server can log onto a client machine with `ypbind` pointing at the correct server.

All that is needed is for the user is to access a home directory. This can be done in a number of ways. We will describe one implementation using NFS.

We assume that all the home directories are on a single server with the following IP address 10.0.0.1

All the clients are on the 10.0.0.0/8 network.

**On the NFS server**

Edit `/etc/exports` and add

```
/home 10.0.0.1/8(rw)
```

Notice that `root_squash` will apply automatically.

**On the client**

Edit `/etc/fstab` and add

```
10.0.0.1:/home /home defaults 0 0
```

2.5 Basic NIS Administration

With the latest versions of `ypserv` a number of default maps are created using source files in `/etc`. It is possible to alter the `YPPWDDIR` and `YPSRCDIR` variables in the Makefile to build maps from alternative files from custom locations.

Updates are made with the Makefile in `/var/yp`. The targets are `all, passwd, group` ...

Copy the new maps to `/var/yp/linis` and run `yppush` to update the slave servers:

```
yppush MAP_NAME
```
Additional Commands

<table>
<thead>
<tr>
<th>Command</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>ypcat</td>
<td>get values from a database, for example <code>ypcat passwd</code></td>
</tr>
<tr>
<td>ypwhich</td>
<td>return the name of the NIS server on the network</td>
</tr>
</tbody>
</table>

3. LDAP Configuration

3.1 What is LDAP

LDAP stands for Lightweight Directory Access Protocol. The protocol allows access to data in a tree-like structure using attributes. LDAP can be thought of as a specialised database which handles trees. Since directories are also trees, navigating LDAP fields is like navigating a directory. Added to this LDAP has been designed mainly for optimal access. This clarifies the words Directory and Access.

The Distinguished Name

An item in the database can be referenced using a unique Distinguished Name (dn). This is similar to a file’s full path in a directory. Each intermediate subfolder is called a Relative Distinguished Name.

More Terminology

<table>
<thead>
<tr>
<th>Term</th>
<th>Definition</th>
</tr>
</thead>
<tbody>
<tr>
<td>DIT</td>
<td>The Data Information Tree</td>
</tr>
<tr>
<td>DN</td>
<td>Distinguished Name</td>
</tr>
<tr>
<td>RDN</td>
<td>Relative Distinguished Name</td>
</tr>
<tr>
<td>LDIF</td>
<td>LDAP Data Interchange Format</td>
</tr>
</tbody>
</table>
Attributes:

dc  Domain Component  
cn  Common Name  
c  Country  
l  Location  
o  Organisation  
ou  Organisational Unit  
sn  Surname  
st  State  
uid  User id  

3.2 OpenLDAP server configuration

The server is called *slapd* (Standalone LDAP daemon) and its configuration file is:

```
/etc/openldap/slapd.conf
```

We will cover each section of this file in more detail.

**Importing schemas**

There is an `include` clause in *slapd.conf* which tells the LDAP server which schemas should be loaded.

We need at least the following:

```bash
include /etc/openldap/schema/core.schema
include /etc/openldap/schema/misc.schema
include /etc/openldap/schema/cosine.schema
include /etc/openldap/schema/nis.schema
include /etc/openldap/schema/inetorgperson.schema
```

**Database Definition**

Available DBMs (Database Managers) are *ldbm* or the more recent *bdb*.

We will use *bdb*:

```
database bdb
```
You need to specify the root or base for the LDAP directory, as well as the directory where the database file will be kept. This is done below:

```
suffix    "dc=example,dc=com"
directory /var/lib/ldap/
```

The following lines are only needed when modifying the LDAP server online. You can then specify an administrator username/password. Use the `slappasswd` to generate an encrypted hash (see 3.4 Migrating System Files to LDAP):

```
rootdn    "cn=Manager,dc=example,dc=com"
rootpw    {SSHA}KiXS5htbnVEQp70rjoteQZHHICs0krBO
```

### 3.3 Client configuration files

There are two configuration files called ldap.conf. Here is what they do:

- The `/etc/ldap.conf` file is used by the nss_ldap and pam_ldap modules
- The file `/etc/openldap/ldap.conf` is used by the tools `ldapsearch` and `ldapadd`

For example, to save time typing:

```
ldapsearch -b "dc=example,dc=com" -x
```

you can add the next lines to `/etc/openldap/ldap.conf`

```
BASE       dc=example, dc=com
HOST       127.0.0.1
```

So far we have configured `slapd` and the configuration file for `ldapsearch` in particular. Once we have populated an LDAP directory we will be able to test our setup by typing:

```
ldapsearch -x
```

### 3.4 Migrating System Files to LDAP

There are two methods available to populate an LDAP directory.

- If the ldap daemon `slapd` is stopped, we can do an offline update using `slapadd`
- While `slapd` is running, it is possible to perform an online update using `ldapadd` or `ldapmodify`
We will also use migration tools which can be downloaded from http://www.padl.com/OSS/MigrationTools.html.

Creating LDAP directories offline

We are going to work in the directory containing the LDAP migration Perl scripts which we have downloaded from www.padl.com.

Notice: Some distributions may include the migration tools with the LDAP server package.

You should have the following files:

- migrate_automount.pl
- CVSVersionInfo.txt
- Make.rules
- MigrationTools.spec
- README
- ads
- migrate_netgroup_byhost.pl
- migrate_netgroup_byuser.pl
- migrate_networks.pl
- migrate_passwd.pl
- migrate_profile.pl
- migrate_protocols.pl
- migrate_rpc.pl
- migrate_services.pl
- migrate_slapd_conf.pl
- migrate_base.pl
- migrate_common.ph
- migrate_fstab.pl
- migrate_group.pl
- migrate_hosts.pl
- migrate_netgroup.pl
- migrate_netgroup_byhost.pl
- migrate_netgroup_byuser.pl
- migrate_all_netinfo_ofline.sh
- migrate_all_netinfo_online.sh
- migrate_all_nis_offline.sh
- migrate_all_nis_online.sh
- migrate_all_nisplus_offline.sh
- migrate_all_nisplus_online.sh
- migrate_all_offline.sh
- migrate_all_online.sh

First edit `migrate_common.ph` and change the $DEFAULT_BASE variable to:

$DEFAULT_BASE = "dc=example,dc=com";

NOTICE

When migrating the /etc/passwd file one can either use shadow passwords or not. When using shadow passwords an added objectClass called shadowAccount is used in the LDAP record and there is no need to migrate the shadow password file.

We create our first LDIF file called `base.ldif` to serve as our root:

```
/migrate_base.pl > base.ldif
```

This flat file will be converted into bdb (or ldbm) files stored in /var/lib/ldap as follows:

```
slapadd -v < base.ldif
```
We next choose to migrate the password without shadow passwords as follows:

```
./migrate_passwd.pl /etc/passwd passwd.ldif
```

The entries in `passwd.ldif` should look like this:

```
dn: uid=test,ou=People,dc=example,dc=com
uid: test
cn: test
objectClass: account
objectClass: posixAccount
objectClass: top
userPassword: {crypt}$1$FGrRfa0u$lo5XwA9xxsmjboNB2Z361
loginShell: /bin/bash
uidNumber: 505
gidNumber: 506
homeDirectory: /home/test
```

Now let's add this LDIF file to our LDAP directory:(remember that LDAP is stopped so we are still offline)

```
slapadd -v -l passwd.ldif  or
slapadd -v < passwd.ldif
```

**NOTICE:**
Make sure all the files in `/var/lib/ldap` belong to user `ldap`.

**TESTING:**

Restart the LDAP server:

```
/etc/init.d/ldap restart
```

Search all the entries in the directory:

```
ldapsearch -x
```

If the `ldap` server does not respond, or the result from `ldapsearch` is empty, it is possible to show the content of the LDAP databases in `/var/lib/ldap` with the `slapcat` command.
Creating LDAP Directories Online

The LDAP server can be updated online, without having to shut the ldap service down. For this to work we must specify a rootdn and a rootpw in /etc/openldap/slapd.conf.

The password is generated from the command line as follows

```
sldappasswd
New password:
Re-enter new password:
{SSHA}XyZmHH1RlnSVXTj87UvxOAOCA8oxNCT
```

We next choose the rootdn in /etc/openldap/slapd.conf to be

```
rootdn          "cn=Manager,dc=example,dc=com"
rootpw          {SSHA}XyZmHH1RlnSVXTj87UvxOAOCA8oxNCT
```

The next line will update the LDAP entries

```
ldapmodify -f passwd.ldif -x -D "dc=example,dc=com" -W
Enter LDAP Password:
```

3.5 LDAP Authentication Scheme

Server Configuration

We assume that the LDAP server has been configured as above.

The passwords in the LDAP directory can also be updated online with the ldappasswd command.

The next line will update the password for user tux on the LDAP server.

```
ldappasswd -D "cn=Manager,dc=example,dc=com" -S -x -W "uid=tux,ou=People,dc=example,dc=com"
```

The -S switch is used to configure a new password.

We assume that the IP address for the server is 10.0.0.1 and that the domain component is "dc=example,dc=com"
You may allow users to change their passwords on the LDAP server as follows:

1. Copy the `passwd` PAM file `/etc/share/doc/nss_ldap-version/pam.d/passwd` to `/etc/pam.d`.

2. Add the following access rule in `/etc/openldap/slapd.conf`:

   ```
   access to attrs=userPassword
   by self write
   by anonymous auth
   by * none
   ```

**Client Configuration**

The clients need to have the `nss_ldap` package installed (some distributions have a separate `pam_ldap` package with the PAM related modules and files). The following files and libraries are installed:

<table>
<thead>
<tr>
<th>File</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td><code>/etc/ldap.conf</code></td>
<td>set the hostname and the domain component of the LDAP server used for authentications</td>
</tr>
<tr>
<td><code>/lib/libnss_ldap-2.3.2.so</code></td>
<td>an ldap module for the NameService Switch</td>
</tr>
<tr>
<td><code>/lib/security/pam_ldap.so</code></td>
<td>the PAM ldap module</td>
</tr>
<tr>
<td><code>/usr/lib/libnss_ldap.so</code></td>
<td>a symbolic link to <code>/lib/libnss_ldap-2.3.2.so</code></td>
</tr>
<tr>
<td><code>/usr/share/doc/nss_ldap-207/pam.d</code></td>
<td>sample files for programs using PAM</td>
</tr>
</tbody>
</table>

If we don't use SSL certificates then `/etc/ldap.conf` is as follows:

**The `/etc/ldap.conf` file**

```
host 10.0.0.1
base dc=example,dc=com
ssl no
pam_password md5
```

Next in `/etc/pam.d` replace the file called `login` with `/usr/share/doc/nss_ldap-207/pam.d/login`. This will tell the authentication binary `/bin/login` to use the `pam_ldap.so` module.

Finally the `/etc/nsswitch.conf` needs to have the following line:

```
passwd ldap files
```

Check the `/var/log/ldap/ldap.log` file on the server to follow the authentication process.
4. PAM Authentication

Services or applications which need authentication can use the pluggable authentication module (PAM) mechanism which offers a modular approach to the authentication process. For example, if a new hardware authentication scheme is added to a system, using smart cards or prime number generators, and if corresponding PAM library modules are available for this new scheme, then it is possible to modify existing services to use this new authentication scheme.

4.1 PAM Aware Applications

Services which use pluggable authentication modules have been compiled with `libpam`. For example `sshd` is such a service:

```
ldd `which sshd` | grep pam
libpam.so.0 => /lib/libpam.so.0 (0x00941000)
```

These applications will scan the PAM configuration files which in turn tell the application how the authentication will take place.
4.2 PAM Configuration

PAM configuration is controlled with the single file `/etc/pam.conf`. This file contains a list of services and a set of instructions, as follows:

```
  service  type  control  module-path  module-arguments
```

However, if the directory `/etc/pam.d` exists then `pam.conf` is ignored and each service is configured through a separate file in `pam.d`. These files are similar to `pam.conf` except that the `service` name is dropped:

```
type  control  module-path  module-arguments
```

- **type**: defines the “management group type”. PAM modules are classified into four management groups which define different aspects of the authentication process:
  - **account**: check the validity of the account (eg. does the users have a UNIX account, is the user authorised to use the application ...)
  - **auth**: the authentication method. This points to a module(s) responsible for the challenge-response
  - **password**: defines how to change user passwords, if at all
  - **session**: modules that are run before and after a service is granted

- **control**: defines what action to take if the module fails. The simple controls are:
  - **requisite**: a failure of the module results in the immediate termination of the authentication process
  - **required**: a failure of the module will result in the termination of the authentication once all the other modules of the same type have been executed
  - **sufficient**: success of the module is sufficient except if a prior `required` module has failed
  - **optional**: success or failure of this module are not taken into account unless it is the only requirement of its type

- **module-path**: the path to a PAM module (usually in `/lib/security`)

- **module-arguments**: list of arguments for a specific module
1. Iptables/Ipc

What is a Packet Filter?

A packet filter is a piece of software which looks at the header of packets as they pass through, and decides the fate of the entire packet. It might decide to DROP the packet (i.e., discard the packet as if it had never received it), ACCEPT the packet (i.e., let the packet go through), or something more complicated. - from the “Packet Filtering HOWTO” by Rusty Russell

For more in depth information see the HOWTOs at www.netfilter.org.

In this section we introduce the **iptables** concepts of chains, tables and targets. We then look at some examples to illustrate network address translation (NAT) as well as the special cases of masquerading and transparent redirections.

1.1 The Chains

A chain is a list of rules which by considering criteria found in the packet's header will make decisions about the type of action to take (target). There are five chains corresponding to different stages in the netfilter framework: PREROUTING, INPUT, FORWARD, POSTROUTING and OUTPUT.

Below is a diagram of the progression of a packet through the kernel netfilter framework:
1.2 The Tables

There are three built-in tables (the IP Tables) which allow to carry out different tasks as listed below.

<table>
<thead>
<tr>
<th>filter:</th>
<th>this is the default table and the packets are never altered. Packets are available from the following chains:</th>
</tr>
</thead>
<tbody>
<tr>
<td>INPUT</td>
<td>for packets coming into the box itself</td>
</tr>
<tr>
<td>OUTPUT</td>
<td>for locally-generated packets</td>
</tr>
<tr>
<td>FORWARD</td>
<td>for packets being routed through the box (check the value of /proc/sys/net/ipv4/ip_forward)</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>nat:</th>
<th>this table only deals with network address translations (NAT) it is consulted when a packet creating a new connection is encountered. Packet headers connected with routing can be altered here. The following chains are considered:</th>
</tr>
</thead>
<tbody>
<tr>
<td>PREROUTING</td>
<td>alters the packets as they come in</td>
</tr>
<tr>
<td>POSTROUTING</td>
<td>alters packets as they go out</td>
</tr>
<tr>
<td>OUTPUT</td>
<td>alters locally generated packets before routing</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>mangle:</th>
<th>used for specialized packet alterations. Targets in this table allow the TOS or TTL field to be modified.</th>
</tr>
</thead>
<tbody>
<tr>
<td>Until kernel 2.4.17 it could only interact with two chains:</td>
<td></td>
</tr>
<tr>
<td>PREROUTING</td>
<td>for altering incoming packets before routing</td>
</tr>
<tr>
<td>OUTPUT</td>
<td>for altering locally-generated packets before routing</td>
</tr>
<tr>
<td>Since kernel 2.4.18, the three other chains are also supported:</td>
<td></td>
</tr>
<tr>
<td>INPUT</td>
<td>for packets coming into the box itself</td>
</tr>
<tr>
<td>FORWARD</td>
<td>for altering packets being routed through the box</td>
</tr>
<tr>
<td>POSTROUTING</td>
<td>for altering packets as they are about to go out</td>
</tr>
</tbody>
</table>

1.3 The Targets

The part of the filtering rule which determines what action to take if the rule is matched is called a target and is preceded by a -j flag (jump). Here is an overview of available targets for a given table:
all tables: ACCEPT, REJECT, DROP, LOG, ULOG, TCPMSS, MIRROR
filter: (nothing individual to this chain)
nat: DNAT, SNAT, MASQUERADE, REDIRECT
mangle: TOS, MARK, DSCP, ECN

There are more targets, but they come as part of additional extension kernel modules.

1.4 Example Rules

1. Example filter rules:

Drop incoming icmp-request as well as outgoing icmp-reply packets

```
iptables -A INPUT -p icmp --icmp-type echo-request -j DROP
iptables -A OUTPUT -p icmp --icmp-type echo-reply -j DROP
```

Notice: The protocol extension flags allow you to specify more information about a specific protocol. In the case of TCP packets for example you may have:

```
-p tcp -tcp-flags ALL SYN,ACK
```

ALL stands for SYN ACK FIN RST URG and PSH. This rule says that all flags must be examined and of those, if the SYN and ACK flags are set, the rule is true.

2. Example Destination Network Address Translation (DNAT):

All requests on port 80 for host 192.168.3.100 are redirected to the host 10.1.1.1 on port 80:

```
iptables -t nat -A PREROUTING -p tcp -i eth0 -d 192.168.3.100 --dport 80 -j DNAT --to 10.1.1.1:80
```

3. Example Source Network Address Translation (SNAT):

The SNAT target is used to change the Source Address. For example, in the case where a router switches from the address on all outgoing packets leaving through ppp0 to its own (public) IP address. The line would look like this:

```
iptables -t nat -A POSTROUTING -o ppp0 -s 192.168.3.0/24 -d 0/0 --j SNAT --to ROUTER_IP
```

This rule can also be written using the MASQUERADE target:

```
iptables -t nat -A POSTROUTING -o ppp0 -s 192.168.3.0/24 -d 0/0 -j MASQUERADE
```
4. Example Redirection:

A redirection is a special case of DNAT where the `--to host` is the same host. For example if a proxy server is running on a router, all requests through port 80 can be PRE-routed through port 3128 with:

```
iptables -A PREROUTING -t nat -i eth0 -p tcp --dport 80 -j REDIRECT --to-port 3128
```

**TASK:** At this stage if you want to implement a transparent proxy with the previous redirection rule you will have to change the configuration file `squid.conf` and add the following:

```
httpd_accel_host virtual
httpd_accel_port 80
httpd_accel_with_proxy on
httpd_accel_uses_host_header on
```

Remember that if you have implemented an authentication scheme with squid you may have to disable it for the transparent proxy to work.

2. Differences with `ipchains`

Some of the main improvement over `ipchains`:

With `iptables`, each filtered packet is only processed using rules from one chain rather than multiple chains. In other words, a FORWARD packet coming into a system using `ipchains` would have to go through the INPUT, FORWARD, and OUTPUT chains in order to move along to its destination. However, `iptables` only sends packets to the INPUT chain if they are destined for the local system and only sends them to the OUTPUT chain if the local system generated the packets. For this reason, you must be sure to place the rule designed to catch a particular packet in the correct chain that will actually see the packet.

The advantage is that you now have finer-grained control over the disposition of each packet. If you are attempting to block access to a particular website, it is now possible to block access attempts from clients running on hosts which use your host as a gateway. An OUTPUT rule which denies access will no longer prevent access for hosts which use your host as a gateway.

**Additional Matching Extensions**

Matching extensions are implemented in `iptables` as modules. Modules are invoked with the `-m` switch.

For example the `state` module makes it possible to distinguish new packets and packets from an established connect. The packet is tested for a matching `state`. Particular state
values are NEW, ESTABLISHED, RELATED or INVALID.

<table>
<thead>
<tr>
<th>Command</th>
</tr>
</thead>
<tbody>
<tr>
<td><code>iptables -A INPUT -p tcp -m state --state ESTABLISHED -j ACCEPT</code></td>
</tr>
<tr>
<td><code>iptables -A OUTPUT -p tcp -m state --state NEW,ESTABLISHED -j ACCEPT</code></td>
</tr>
</tbody>
</table>

Matching extension modules are listed below.

<table>
<thead>
<tr>
<th>Module</th>
<th>Description</th>
<th>Option (example)</th>
</tr>
</thead>
<tbody>
<tr>
<td>connrate</td>
<td>matches the current connection rate</td>
<td><code>--connrate [] [from]:[to]</code></td>
</tr>
<tr>
<td>dstlimit</td>
<td>This module allows you to limit the packet per second (pps) rate on a per destination IP or per destination port base</td>
<td><code>--dstlimit avg</code></td>
</tr>
<tr>
<td>icmp</td>
<td>this extension is loaded if <code>--protocol icmp</code> is specified</td>
<td><code>--icmptype [] [] typename</code></td>
</tr>
<tr>
<td>iprange</td>
<td>specify a range of IPs</td>
<td><code>--src-range IP-IP</code></td>
</tr>
<tr>
<td>length</td>
<td>matches the length of the packet</td>
<td><code>--length length</code></td>
</tr>
<tr>
<td>mac</td>
<td>match the MAC source</td>
<td><code>--mac-source [] [] address</code></td>
</tr>
<tr>
<td>state</td>
<td>determine the state of a packet (NEW,ESTABLISHED,RELATED,INVALID)</td>
<td><code>--state state</code></td>
</tr>
</tbody>
</table>

3. Security Tools

3.1 SSH

For a first description of the `ssh` client and `sshd` server see the section on “Basic Security” in the lpi-manuals document for LPI 102. For an in depth presentation see the Internet draft “The SSH (Secure Shell) Remote Login Protocol” at http://www.free.lp.se/fish/rfc.txt.

This section covers the server configuration file and briefly discusses other mechanisms that the SSH protocol offers such as X11 forwarding and port forwarding.
sshd_config overview

<table>
<thead>
<tr>
<th></th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Port 22</td>
<td>Specify which port to listen on. Multiple “Port” options can be used</td>
</tr>
<tr>
<td>Protocol 2,1</td>
<td>Specify version 1 or version 2 SSH protocol. Can be a comma separated list. If both are supplied, they are tried in the order presented.</td>
</tr>
<tr>
<td>DenyUsers [USER]@HOST</td>
<td>Deny users from a specific host. Wild cards such as * can be used</td>
</tr>
<tr>
<td>IgnoreRhosts yes/no</td>
<td>Default is yes – Ignore the ~/.rhosts and ~/.shosts files</td>
</tr>
<tr>
<td>PermitEmptyPasswords yes/no</td>
<td>Default is no – Allow login with an empty passwords when password authentication is allowed</td>
</tr>
<tr>
<td>PermitRootLogin yes/no</td>
<td>Allow or disallow root access</td>
</tr>
<tr>
<td>X11Forwarding yes/no</td>
<td>Instructs the remote end to route X11 traffic back through the ssh tunnel to the user’s X session. Unless disabled, the xauth settings will be transferred in order to properly authenticate remote X applications</td>
</tr>
</tbody>
</table>

Port Forwarding

It is possible to do port forwarding with the SSH client. This is often used to provide a simple mechanism to encrypt a connection. For example one can open a local (-L) port (1234) pointing to the remote host (www.google.com) on another port (80) as follows:

```
ssh -L 1234:www.google.com:80 127.0.0.1
```

Quick VPN

This is a user-space VPN as opposed to other types of VPNs which are kernel based.

```
/usr/sbin/pppd noauth pty \
"ssh SOME_HOST -l root '/usr/sbin/pppd notty noauth 192.168.0.1:192.168.0.2'" \
192.168.0.2:192.168.0.1
```
3.2 LSOF

`ls假期 shows open files used by processes. Traditionally it is used to list PIDs of processes running on a given directory:

```
ls假期 +D DIRECTORY
```

`ls假期` will output the following information:

<table>
<thead>
<tr>
<th>NAME</th>
<th>name of the process</th>
</tr>
</thead>
<tbody>
<tr>
<td>PID</td>
<td>process ID</td>
</tr>
<tr>
<td>USER</td>
<td>name of the user to whom the process belongs</td>
</tr>
<tr>
<td>FD</td>
<td>File descriptor (e.g u = read write, r = read, w = write)</td>
</tr>
<tr>
<td>TYPE</td>
<td>The file type (e.g REG = regular file)</td>
</tr>
<tr>
<td>DEVICE</td>
<td>Major/Minor number (e.g 3,16 =/dev/hda16 )</td>
</tr>
<tr>
<td>SIZE</td>
<td>Size or offset of the file</td>
</tr>
<tr>
<td>NODE</td>
<td>Inode of the file</td>
</tr>
<tr>
<td>NAME</td>
<td>The name of the file</td>
</tr>
</tbody>
</table>

`ls假期` can also be used to display network sockets. For example the following line will list all internet connections:

```
ls假期 -i
```

You can also list connections to a single host:

```
ls假期 -i @HOST
```

For example if a host TOFFY is connected to your localhost on port 1234, the following would display information about the connection:

```
ls假期 -i @TOFFY:1234
```
3.3 NETSTAT

`netstat` is used to print network connections, routing tables, etc.

Main options are:

- `-r` display routing tables
- `-l` only listening services
- `-C` display route cache
- `--inet` restrict to network sockets

Protocol types are:

- `-t` select tcp
- `-u` select udp

3.4 TCPDUMP

`tcpdump` dumps traffic on a network.

From the man page:

◆ The TCP Packet

The general format of a tcp protocol line is:

```
src > dst: flags data-seqno ack window urgent options
```

**Src and dst** - the source and destination IP addresses and ports.

**Flags** - some combination of S (SYN), F (FIN), P (PUSH) or R (RST) or a single ‘.’ (no flags).

**Data-seqno** - describes the portion of sequence space covered by the data in this packet (see example below).

**Ack** - sequence number of the next data expected in the other direction on this connection.

**Window** - the number of bytes of receive buffer space available in the other direction on this connection.

**Urg** - indicates there is ‘urgent’ data in the packet.

**Options** - tcp options enclosed in angle brackets (e.g., `<mss 1024>`)
Capturing TCP packets with particular flag combinations (e.g. SYN-ACK, URG-ACK, etc.)

There are 8 bits in the control bits section of the TCP header:

```
CWR | ECE | URG | ACK | PSH | RST | SYN | FIN
```

Let’s assume that we want to watch packets used in establishing a TCP connection. Recall the structure of a TCP header without options:

```
<p>|          source port          |       destination port        |
|--------------------------------|</p>
<table>
<thead>
<tr>
<th>sequence number</th>
</tr>
</thead>
<tbody>
<tr>
<td>acknowledgment number</td>
</tr>
<tr>
<td>--------------------------------</td>
</tr>
<tr>
<td>HL</td>
</tr>
<tr>
<td>--------------------------------</td>
</tr>
<tr>
<td>TCP checksum</td>
</tr>
</tbody>
</table>
```

A TCP header usually holds 20 octets of data, unless options are present. The first line of the graph contains octets 0 - 3, the second line shows octets 4 - 7 etc.

Starting to count with 0, the relevant TCP control bit are contained in octet 13:

```
<p>|          source port          |       destination port        |
|--------------------------------|</p>
<table>
<thead>
<tr>
<th>sequence number</th>
</tr>
</thead>
<tbody>
<tr>
<td>acknowledgment number</td>
</tr>
<tr>
<td>--------------------------------</td>
</tr>
<tr>
<td>HL</td>
</tr>
<tr>
<td>--------------------------------</td>
</tr>
<tr>
<td></td>
</tr>
</tbody>
</table>
```

Let’s have a closer look at octet no. 13:

```
-------------------
|C|E|U|A|P|R|S|F|
-------------------
```

These are the TCP control bits we are interested in. We have numbered the bits in this octet from 0 to 7, right to left, so the PSH bit is bit number 3, while the URG bit is number 5.

Recall that we want to capture packets with only SYN set. Let’s see what happens to octet 13 if a TCP datagram arrives with the SYN bit set in its header:
Looking at the control bits section we see that only bit number 1 (SYN) is set.

Assuming that octet number 13 is an 8-bit unsigned integer in network byte order, the binary value of this octet is

```
<table>
<thead>
<tr>
<th>C</th>
<th>E</th>
<th>U</th>
<th>A</th>
<th>P</th>
<th>R</th>
<th>S</th>
<th>F</th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>1</td>
<td>0</td>
<td>0</td>
</tr>
</tbody>
</table>
```

and its decimal representation is

```
7 6 5 4 3 2 1 0
0*2 + 0*2 + 0*2 + 0*2 + 0*2 + 1*2 + 0*2 = 2
```

We’re almost done, because now we know that if only SYN is set, the value of the 13th octet in the TCP header, when interpreted as a 8-bit unsigned integer in network byte order, must be exactly 2.

This relationship can be expressed as

```
tcp[13] == 2
```

### 3.5 NMAP

**nmap** is the network exploration tool and security scanner.

The scanner makes use of the fact that a closed port should (according to RFC 793) send back an RST. In the case if a SYN scan, connections that are half opened are immediately close by nmap by sending an RST itself.

The main scan types are:

**SYN or Half-open: -sS**

nmap will send a synchronisation packet SYN asking for a connection. If the remote host send a RST/ACK it is assumed that the port is closed. If the remote host sends a SYN/ACK this indicates that the port is listening.

**UDP: -sU**

UDP is connectionless. So there is no need for a 3 way handshake as with TCP. If a port is closed the server will send back a ICMP PORT UNREACHABLE. One then deduces
that all the other ports are open (not reliable in the case were ICMP messages are blocked).

**TCP NULL: -sN**
TCP packet with no flags set. Closed port will send a RST when receiving these packets (except with MS Windows).

**TCP Xmas: -sX**
TCP packet with the FIN+URG+PUSH flags set. The remote host should send back a RST for all closed ports when receiving a Xmas packet.
GNU Free Documentation License

Version 1.2, November 2002

Copyright (C) 2000, 2001, 2002 Free Software Foundation, Inc.
59 Temple Place, Suite 330, Boston, MA 02111-1307 USA
Everyone is permitted to copy and distribute verbatim copies of this license document, but changing it is not allowed.

0. PREAMBLE

The purpose of this License is to make a manual, textbook, or other functional and useful document "free" in the sense of freedom: to assure everyone the effective freedom to copy and redistribute it, with or without modifying it, either commercially or non-commercially. Secondly, this License preserves for the author and publisher a way to get credit for their work, while not being considered responsible for modifications made by others.

This License is a kind of "copyleft", which means that derivative works of the document must themselves be free in the same sense. It complements the GNU General Public License, which is a copyleft license designed for free software.

We have designed this License in order to use it for manuals for free software, because free software needs free documentation: a free program should come with manuals providing the same freedoms that the software does. But this License is not limited to software manuals; it can be used for any textual work, regardless of subject matter or whether it is published as a printed book. We recommend this License principally for works whose purpose is instruction or reference.

1. APPLICABILITY AND DEFINITIONS

This License applies to any manual or other work, in any medium, that contains a notice placed by the copyright holder saying it can be distributed under the terms of this License. Such a notice grants a world-wide, royalty-free license, unlimited in duration, to use that work under the conditions stated herein. The "Document", below, refers to any such manual or work. Any member of the public is a licensee, and is addressed as "you". You accept the license if you copy, modify or distribute the work in a way requiring permission under copyright law.

A "Modified Version" of the Document means any work containing the Document or a portion of it, either copied verbatim, or with modifications and/or translated into another language.

A "Secondary Section" is a named appendix or a front-matter section of the Document that deals exclusively with the relationship of the publishers or authors of the Document to the Document's overall subject (or to related matters) and contains nothing that could fall directly within that overall subject. (Thus, if the Document is in part a textbook of mathematics, a Secondary Section may not explain any mathematics.) The relationship could be a matter of historical connection with the subject or with related matters, or of legal, commercial, philosophical, ethical or political position regarding them.

The "Invariant Sections" are certain Secondary Sections whose titles are designated, as being those of Invariant Sections, in the notice that says that the Document is released under this License. If a section does not fit the above definition of Secondary then it is not allowed to be designated as Invariant. The Document may contain zero Invariant Sections. If the Document does not identify any Invariant Sections then there are none.

The "Cover Texts" are certain short passages of text that are listed, as Front-Cover Texts or Back-Cover Texts, in the notice that says that the Document is released under this License. A Front-Cover Text may be at most 5 words, and a Back-Cover Text may be at most 25 words.

A "Transparent" copy of the Document means a machine-readable copy, represented in a format whose specification is available to the general public, that is suitable for revising the document straightforwardly with generic text editors or (for images composed of pixels) generic paint programs or (for drawings) some widely available drawing editor, and that is suitable for input to text formatters or for automatic translation to a variety of formats suitable for input to text formatters. A copy made in an otherwise Transparent file format whose markup, or absence of markup, has been arranged to thwart or discourage subsequent modification by readers is not Transparent. An image format is not Transparent if used for any substantial amount of text. A copy that is not "Transparent" is called
"Opaque".

Examples of suitable formats for transparent copies include plain ASCII without markup, Texinfo input format, LaTeX input format, SGML or XML using a publicly available DTD, and standard-conforming simple HTML, PostScript or PDF designed for human modification. Examples of transparent image formats include PNG, XCF and JPG. Opaque formats include proprietary formats that can be read and edited only by proprietary word processors, SGML or XML, for which the DTD and/or processing tools are not generally available, and the machine-generated HTML, PostScript or PDF produced by some word processors for output purposes only.

The "Title Page" means, for a printed book, the title page itself, plus such following pages as are needed to hold, legibly, the material this License requires to appear in the title page. For works in formats which do not have any title page as such, "Title Page" means the text near the most prominent appearance of the work's title, preceding the beginning of the body of the text.

A section "Entitled XYZ" means a named subunit of the Document whose title either is precisely XYZ or contains XYZ in parentheses following text that translates XYZ in another language. (Here XYZ stands for a specific section name mentioned below, such as "Acknowledgements", "Dedications", "Endorsements", or "History"). To "Preserve the Title" of such a section when you modify the Document means that it remains a section "Entitled XYZ" according to this definition.

The Document may include Warranty Disclaimers next to the notice which states that this License applies to the Document. These Warranty Disclaimers are considered to be included by reference in this License, but only as regards disclaiming warranties: any other implication that these Warranty Disclaimers may have is void and has no effect on the meaning of this License.

2. VERBATIM COPYING

You may copy and distribute the Document in any medium, either commercially or non-commercially, provided that this License, the copyright notices, and the license notice saying this License applies to the Document are reproduced in all copies, and that you add no other conditions whatsoever to those of this License. You may not use technical measures to obstruct or control the reading or further copying of the copies you make or distribute. However, you may accept compensation in exchange for copies. If you distribute a large enough number of copies you must also follow the conditions in section 3.

You may also lend copies, under the same conditions stated above, and you may publicly display copies.

3. COPYING IN QUANTITY

If you publish printed copies (or copies in media that commonly have printed covers) of the Document, numbering more than 100, and the Document's license notice requires Cover Texts, you must enclose the copies in covers that carry, clearly and legibly, all these Cover Texts: Front-Cover Texts on the front cover, and Back-Cover Texts on the back cover. Both covers must also clearly and legibly identify you as the publisher of these copies. The front cover must present the full title with all words of the title equally prominent and visible. You may add other material on the covers in addition. Copying with changes limited to the covers, as long as they preserve the title of the Document and satisfy these conditions, can be treated as verbatim copying in other respects.

If the required texts for either cover are too voluminous to fit legibly, you should put the first ones listed (as many as fit reasonably) on the actual cover, and continue the rest onto adjacent pages.

If you publish or distribute Opaque copies of the Document numbering more than 100, you must either include a machine-readable Transparent copy along with each Opaque copy, or state in or with each Opaque copy a computer-network location from which the general network-using public has access to download using public-standard network protocols a complete Transparent copy of the Document, free of added material. If you use the latter option, you must take reasonably prudent steps, when you begin distribution of Opaque copies in quantity, to ensure that this Transparent copy will remain thus accessible at the stated location until at least one year after the last time you distribute an Opaque copy (directly or through your agents or retailers) of that edition to the public.

It is requested, but not required, that you contact the authors of the Document well before redistributing any large number of copies, to give them a chance to provide you with an updated version of the Document.
4. MODIFICATIONS

You may copy and distribute a Modified Version of the Document under the conditions of sections 2 and 3 above, provided that you release the Modified Version under precisely this License, with the Modified Version filling the role of the Document, thus licensing distribution and modification of the Modified Version to whoever possesses a copy of it. In addition, you must do these things in the Modified Version:

- **A.** Use in the Title Page (and on the covers, if any) a title distinct from that of the Document, and from those of previous versions (which should, if there were any, be listed in the History section of the Document). You may use the same title as a previous version if the original publisher of that version gives permission.

- **B.** List on the Title Page, as authors, one or more persons or entities responsible for authorship of the modifications in the Modified Version, together with at least five of the principal authors of the Document (all of its principal authors, if it has fewer than five), unless they release you from this requirement.

- **C.** State on the Title page the name of the publisher of the Modified Version, as the publisher.

- **D.** Preserve all the copyright notices of the Document.

- **E.** Add an appropriate copyright notice for your modifications adjacent to the other copyright notices.

- **F.** Include, immediately after the copyright notices, a license notice giving the public permission to use the Modified Version under the terms of this License, in the form shown in the Addendum below.

- **G.** Preserve in that license notice the full lists of Invariant Sections and required Cover Texts given in the Document's license notice.

- **H.** Include an unaltered copy of this License.

- **I.** Preserve the section Entitled "History", Preserve its Title, and add to it an item stating at least the title, year, new authors, and publisher of the Modified Version as given on the Title Page. If there is no section Entitled "History" in the Document, create one stating the title, year, authors, and publisher of the Document as given on its Title Page, then add an item describing the Modified Version as stated in the previous sentence.

- **J.** Preserve the network location, if any, given in the Document for public access to a Transparent copy of the Document, and likewise the network locations given in the Document for previous versions it was based on. These may be placed in the "History" section. You may omit a network location for a work that was published at least four years before the Document itself, or if the original publisher of the version it refers to gives permission.

- **K.** For any section Entitled "Acknowledgements" or "Dedications", Preserve the Title of the section, and preserve in the section all the substance and tone of each of the contributor acknowledgements and/or dedications given therein.

- **L.** Preserve all the Invariant Sections of the Document, unaltered in their text and in their titles. Section numbers or the equivalent are not considered part of the section titles.

- **M.** Delete any section Entitled "Endorsements". Such a section may not be included in the Modified Version.

- **N.** Do not retitle any existing section to be Entitled "Endorsements" or to conflict in title with any Invariant Section.

- **O.** Preserve any Warranty Disclaimers.

If the Modified Version includes new front-matter sections or appendices that qualify as Secondary Sections and contain no material copied from the Document, you may at your option designate some or all of these sections as invariant. To do this, add their titles to the list of Invariant Sections in the Modified Version's license notice. These titles must be distinct from any other section titles.

You may add a section Entitled "Endorsements", provided it contains nothing but endorsements of your Modified Version by various parties—for example, statements of peer review or that the text has been approved by an organization as the authoritative definition of a standard.

You may add a passage of up to five words as a Front-Cover Text, and a passage of up to 25 words as a Back-Cover Text, to the end of the list of Cover Texts in the Modified Version. Only one passage of Front-Cover Text and one of Back-Cover Text may be added by (or through arrangements made by) any one entity. If the Document already includes a cover text for the same cover, previously added by you or by arrangement made by the same entity you are acting on behalf of, you may not add another; but you may replace the old one, on explicit permission from the previous publisher that added the old one.
GNU FDL License Agreement

The author(s) and publisher(s) of the Document do not by this License give permission to use their names for publicity for or to assert or imply endorsement of any Modified Version.

5. COMBINING DOCUMENTS

You may combine the Document with other documents released under this License, under the terms defined in section 4 above for modified versions, provided that you include in the combination all of the Invariant Sections of all of the original documents, unmodified, and list them all as Invariant Sections of your combined work in its license notice, and that you preserve all their Warranty Disclaimers.

The combined work need only contain one copy of this License, and multiple identical Invariant Sections may be replaced with a single copy. If there are multiple Invariant Sections with the same name but different contents, make the title of each such section unique by adding at the end of it, in parentheses, the name of the original author or publisher of that section if known, or else a unique number. Make the same adjustment to the section titles in the list of Invariant Sections in the license notice of the combined work.

In the combination, you must combine any sections Entitled "History" in the various original documents, forming one section Entitled "History"; likewise combine any sections Entitled "Acknowledgements", and any sections Entitled "Dedications". You must delete all sections Entitled "Endorsements."

6. COLLECTIONS OF DOCUMENTS

You may make a collection consisting of the Document and other documents released under this License, and replace the individual copies of this License in the various documents with a single copy that is included in the collection, provided that you follow the rules of this License for verbatim copying of each of the documents in all other respects.

You may extract a single document from such a collection, and distribute it individually under this License, provided you insert a copy of this License into the extracted document, and follow this License in all other respects regarding verbatim copying of that document.

7. AGGREGATION WITH INDEPENDENT WORKS

A compilation of the Document or its derivatives with other separate and independent documents or works, in or on a volume of a storage or distribution medium, is called an "aggregate" if the copyright resulting from the compilation is not used to limit the legal rights of the compilation's users beyond what the individual works permit. When the Document is included in an aggregate, this License does not apply to the other works in the aggregate which are not themselves derivative works of the Document.

If the Cover Text requirement of section 3 is applicable to these copies of the Document, then if the Document is less than one half of the entire aggregate, the Document's Cover Texts may be placed on covers that bracket the Document within the aggregate, or the electronic equivalent of covers if the Document is in electronic form. Otherwise they must appear on printed covers that bracket the whole aggregate.

8. TRANSLATION

Translation is considered a kind of modification, so you may distribute translations of the Document under the terms of section 4. Replacing Invariant Sections with translations requires special permission from their copyright holders, but you may include translations of some or all Invariant Sections in addition to the original versions of these Invariant Sections. You may include a translation of this License, and all the license notices in the Document, and any Warranty Disclaimers, provided that you also include the original English version of this License and the original versions of those notices and disclaimers. In case of a disagreement between the translation and the original version of this License or a notice or disclaimer, the original version will prevail.

If a section in the Document is Entitled "Acknowledgements", "Dedications", or "History", the requirement (section 4) to Preserve its Title (section 1) will typically require changing the actual title.

9. TERMINATION

You may not copy, modify, sublicense, or distribute the Document except as expressly provided for under this License. Any other attempt to copy, modify, sublicense or distribute the Document is void, and will automatically terminate your rights under this License. However, parties who have received copies, or rights, from you under this License will not have their licenses terminated so long as such
parties remain in full compliance.

10. FUTURE REVISIONS OF THIS LICENSE

The Free Software Foundation may publish new, revised versions of the GNU Free Documentation License from time to time. Such new versions will be similar in spirit to the present version, but may differ in detail to address new problems or concerns. See http://www.gnu.org/copyleft/.

Each version of the License is given a distinguishing version number. If the Document specifies that a particular numbered version of this License "or any later version" applies to it, you have the option of following the terms and conditions either of that specified version or of any later version that has been published (not as a draft) by the Free Software Foundation. If the Document does not specify a version number of this License, you may choose any version ever published (not as a draft) by the Free Software Foundation.