

StockholmOpen.Net Configuration Manual

version 1.0, 2 May 2002

Martin Hedenfalk <mhe@it.kth.se>

Copyright © 2001-2002 Martin Hedenfalk

Permission is granted to make and distribute verbatim copies of this manual provided the copyright notice and this permission notice are preserved on all copies.

Permission is granted to copy and distribute modified versions of this manual under the conditions for verbatim copying, provided that the entire resulting derived work is distributed under the terms of a permission notice identical to this one.

Permission is granted to copy and distribute translations of this manual into another language, under the above conditions for modified versions, except that this permission notice may be stated in a translation approved by the copyright holder.

Table of Contents

1	The basics	1
2	The configuration services	3
2.1	Design of the relay agent	3
2.1.1	Compiling the relay agent	4
2.2	The registration services	4
2.3	Setting up the MAC database	5
2.4	The oasis implementation	5
2.4.1	How oasis is designed	5
2.4.2	Format of the requests	6
2.4.3	Compiling oasis	7
2.4.4	Configuring oasis	7
2.4.5	Starting Oasis	9
2.4.6	PAM	10
2.4.7	The firewall control program	10
2.5	The firewall control daemon	10
2.5.1	Starting fwcd	10
2.5.2	Configuring fwcd	11
3	Step-by-step installation instructions	13

1 The basics

StockholmOpen.Net is an authentication mechanism for a public access network. Internet access is provided by different operators, and users are able to dynamically choose which operator to use, ie, the access network is operator neutral.

Users are required to authenticate to the network in order to gain access. This is done on a secure web page where the user fills in username and password. Before authentication, users are redirected to the login page if the user tries to access another webpage. When the user disconnects from the network, access is again denied. This means that users are required to login each time they want to access the network.

The network consists of two types of services: public services common to the whole network, and provider specific services. The common services are what “holds everything together”. They provide configuration to requesting clients and act as a relay to the provider specific services. The provider specific services are what do the actual authentication, and includes the access server.

The access server is responsible for opening and closing rules in a firewall and thus allowing users to reach the services provided by the provider. These services will typically be access to the global Internet.

The common services in the public network are:

BOOTP relay agent and registration service

This service registers new MAC addresses with a chosen operator and relays DHCP requests to the operators DHCP server based on the choice made by the user. When a new user (with a previously unregistered network card) connects to the registration webpage, he is presented with a choice of upstream providers and the choice is stored in a database. This database is used in following connections to relay DHCP requests.

The common services should be located in the same machine. For each provider, the following services must exist:

DHCP server

This is an ordinary DHCP server. The policy of IP address allocation is not enforced by the system. Either automatic, dynamic or manual allocation could be used. For simplicity, dynamic allocation is recommended.

The DHCP server must be placed outside the open network, in the providers own network, unreachable from unauthenticated users. Otherwise, the functionality of the common relay agent will be disturbed.

Access Server

This service is where users sends his/her credentials in order to login to the network. The user is presented with a web frontend, which is assumed to be secured via an https connection, although this is not a requirement of the system.

Upon successful authentication, the access server communicates with the dynamic firewall to open up a rule, allowing the user access to the upstream network.

The actual authentication need not be performed in the access server. The access server could include a client module that speaks to an authentication server, for example RADIUS or Kerberos.

The access server would then send the user credentials to the authentication server for verification. The authentication server could be common to other services that the operator provides and be located inside the ISP's own network.

Dynamic firewall

The dynamic firewall in the current system is a standard PC running Linux and iptables. The firewall could also be a dedicated router that can be remotely controlled.

The firewall should be configured to drop all packets from a yet unauthenticated user (based on MAC and IP address). If there are certain services that are required for unauthenticated users, such as DNS, there can be static rules that open those services.

2 The configuration services

The configuration services provides configuration to requesting clients and maintains the registration between user and selected provider.

2.1 Design of the relay agent

The BOOTP relay agent is a modified version of ISC's dhcrelay version 3.0. The relay agent has been modified to instead of sending a received DHCP message to one or more servers it now sends the DHCP message to only one server based on the MAC address of the requesting client. This is achieved by looking up the MAC address in a database consisting of mappings between MAC addresses and providers. The database is a PostgreSQL and is accessed through libpq.

The database has the following definitions:

```
CREATE TABLE T_provider (
    provider_id SERIAL,
    provider_name text,
    provider_dhcp inet,
    provider_auth_url text,
    provider_net inet,
    provider_netmask inet,
    primary key (provider_id));
```

```
CREATE TABLE T_macbind (
    mac_addr macaddr,
    provider_id bigint,
    mac_reg_date timestamp,
    primary key (mac_addr));
```

```
CREATE VIEW V_mapping AS SELECT M.mac_addr, P.provider_dhcp,
P.provider_net, P.provider_netmask, P.provider_auth_url FROM T_macbind
AS M,T_provider AS P WHERE M.provider_id = P.provider_id;
```

```
GRANT ALL ON T_provider TO macbind;
GRANT ALL ON T_macbind TO macbind;
GRANT ALL ON V_mapping TO macbind;
```

In the table T_provider:

'provider_id'
A serialnumber identifying the provider that is used to bind mac addresses to it.

'provider_name'
The name of the provider and is also displayed to the users when choosing provider.

'provider_dhcp'
The IP address of the dhcpserver of the provider.

`'provider_auth_url'`
The URL to the providers authentication page.

`'provider_net'`
The IP address of the network that the dhcpserver should hand to a client.

`'provider_netmask'`
The netmask for `'provider_net'`.

In `T_provider` there has to exist one entry with `'provider_name' = "default"` to be used as default when a (MAC address, provider) binding is not available.

In the table `T_macbind`:

`'mac_addr'`
The macaddress of the client.

`'provider_id'`
Refers to the table `T_provider`

`'mac_reg_date'`
A timestamp of when this MAC address last received a DHCP lease. This is used to delete old entries in the database.

The view `V_mapping` is just a way to ease up the queries to the database so that the client only has to make one query to get the (MAC address, provider) binding.

2.1.1 Compiling the relay agent

Unpack the DHCP sources and apply the patch:

```
tar zxf dhcp-3.0.tar.gz
cd dhcp-3.0
patch -p0 < ../dhcp.patch
```

Now configure and make DHCP:

```
./configure
make
make install
```

The relay agent does not need any configuration. All necessary information is retrieved from the MAC database.

For more instructions, see the included documentation in the original ISC DHCP sources.

2.2 The registration services

The registration service lets users choose the provider to use. It is a web page that records the users choice in the MAC database.

When the user wants to authenticate to the network, he enters the registration page. This page looks up the MAC address of the requesting client in the MAC database and redirects the user to the real authentication webpage of the chosen ISP. The registration page also makes it possible for the user to change ISP.

2.3 Setting up the MAC database

PostgreSQL is used for the MAC database.

1. Install PostgreSQL

PostgreSQL is available as convenient packages for many GNU/Linux distributions, including RedHat (+ clones) and Debian.

Alternatively, one can download and compile PostgreSQL from source. The sources are available from <http://www.postgresql.org/>. Instructions for compiling PostgreSQL are included in the sources.

2. Create the database

```
createdb macbind
```

3. Create a user

```
createuser -P macbind
```

The `-P` option will ask you for a password to protect the macbind database. In the current version of the system, this password is hardcoded as `macbind_qwerty` into the access and DHCP relay sources. *This really has to be fixed.*

4. Initialize the database

```
psql -U macbind -f init-macbind.sql macbind
```

5. Configure access permissions

Edit the PostgreSQL access configuration file to allow connections to the macbind database with a password. Add the following line to `/etc/postgresql/pg_hba.conf`:

```
local macbind password
```

2.4 The oasis implementation

This chapter describes the `oasis` implementation.

2.4.1 How oasis is designed

`oasis` is designed as a multi-threaded application. The main thread listens on a UNIX socket for requests. Each request is processed in order, one at a time. When a login request is received, the user is authenticated via PAM (see Section 2.4.6 [PAM], page 10) and, if successful, the configured firewall control program is run to open up a rule for that user (identified by MAC and IP address) in the corresponding firewall.

The requests will typically be sent by the web frontend. In the `www` subdirectory of the source distribution there are PHP scripts that prompts for username and password, retrieves the IP and MAC addresses and sends a request to the access server.

`oasis` uses PAM (Pluggable Authentication Modules) as authentication mechanism. PAM is a highly configurable system, in which one can plug in any authentication module. This system is used on many UN*X systems and there exists many modules, including Kerberos, Radius, plain passwd files and LDAP.

To prevent malicious users from using a valid user authenticated connection after he has disconnected from the network, the rule in the firewall must be closed when a user disconnects. `oasis` has two modes of operation to detect the presence of a user:

1. The access server “probes” the user by sending a “probe” that the user has to respond to. This “probe” will most likely be some sort of ping. To make the system flexible, *oasis* uses a plugin system. Currently, there exists plugins for ARP ping and ICMP ping. The probe plugin to use is configurable per domain.

If the user fails to answer a certain number of such “probes” the user is considered disconnected and the firewall is closed.

Each probe is run in a separate thread to simultaneously probe a (configurable) number of users. It is important to configure the number of simultaneous probes and the maximum number of users allowed in a good way. If too many users are probed simultaneously this could create a flood ping. On the other hand, if too few users are probed, it will take a long time before a user is logged out.

2. The second mode of operation is to use *fwcd*, the firewall control daemon. When a user authenticates to the access server, a request is sent to *fwcd* (running on the firewall) to open up the rule. The connection is secured using SSL. The firewall control daemon then starts to listen on the traffic passed through it. If packets arrive from the users MAC/IP address, he is active and no active action is necessary. When the user is inactive (but still connected) the firewall then starts to probe the user as described in the previous mode until the user starts to send packets again.

This mode eases the load on the network by reducing the amount of “useless” traffic, but also requires that the firewall is able to run the daemon. This is not the case with dedicated routers.

2.4.2 Format of the requests

The request is formatted as a string of ascii characters, terminated with a linefeed - carriage return pair (`\r\n`). The first keyword is the name of the command requested. Each keyword must be separated by space. After the command keyword the arguments follow. An argument is formatted as `OPTION=VALUE`, each argument separated with a linefeed character (`\r`).

```
‘LOGIN’    LOGIN USER=username\rPASS=password\rDOMAIN=domain\rIP=ip-
           address\rMAC=mac-address\r\n
```

```
‘LOGOUT’   LOGOUT\rUSER=username\rDOMAIN=domain\r\n
```

```
‘STATS’    STATS\r\n sends information about all logged in users.
```

```
           STATS\rDOMAIN=domain\r\n sends information about all logged in users in the
           specified domain.
```

```
           STATS\rDOMAIN=domain\rUSER=username\r\n sends information about the
           specified user in that domain.
```

After a request is processed, a reply is sent back. The reply is formatted as `RETURN-CODE ERROR-MESSAGE\r\n`. The return code can be either 0 (successful) or 1 (failed). The `ERROR-MESSAGE` is a string specifying the reason for failure.

See the test clients included in the source package for a more practical example of how requests are sent to the server.

2.4.3 Compiling oasis

`oasis` is distributed as a gzipped tarball. Type `tar xzf oasis-x.x.tar.gz` to unpack the sources (replace `x.x` with the version number). This will create a directory '`oasis-x.x`'.

`oasis` uses a `configure` script to adapt the sources to different platforms. To configure the sources, simply run `./configure` in the top level directory of the source tree. The script accepts some arguments, use `./configure --help` to see them.

To build successfully, `pthread`, `libpcap` and `libnet` are required (and their development headers). If you use a recent distribution, look for the packages `libpthread-dev[el]`, `libpcap-dev[el]` or `libnet-dev[el]`.

The default install prefix is '`/usr/local`', but this can be changed by passing the `--prefix=PATH` argument to `configure`.

After the configuration step, simply type `make` to compile the sources. To install, type `make install` as root.

2.4.4 Configuring oasis

The configuration file is by default installed as '`/usr/local/etc/oasis.conf`'. There is a template in the `etc` subdirectory of the source distribution that you can modify to your needs. The syntax is simple: each directive consists of a keyword, an equal sign and the argument, as in:

```
keyword = argument
```

Comments begin with `#` or `//` and extends to the end of the line. Block comments can be written in C syntax, beginning with `/*` and ending in `*/`.

Valid global directives are:

`probe-interval`

Time between consecutive probe scans of the users. Each scan consists of `probe-max-users` probes.

`probe-max-users`

Maximum number of users to probe in each scan.

`max-missed-probes`

Maximum number of probes a user can miss before the user is considered away and logged out by the system

`probe-timeout`

Timeout, in seconds, before a probe is considered failed and is cancelled. Must be less than `probe-interval`.

`socket-path`

Path to the UNIX socket. Default is `/tmp/oasis`.

`socket-owner`

Owner and group to set for the socket. Owner and group are separated by a dot, as in `root.nobody`.

socket-mode

The permissions to set for the socket. Both the syntax used by `chmod(1)`, as in `"u=rw,g=wx,o=wr"` and octal mode can be used. If octal mode is used, you must surround it in double quotes, as in `"0644"`.

log-facility

What facility to use when logging to syslog. The following arguments are recognised:

`LOG_AUTHPRIV`, `LOG_DAEMON`, `LOG_LOCAL0`, `LOG_LOCAL1`,
`LOG_LOCAL2`, `LOG_LOCAL3`, `LOG_LOCAL4`, `LOG_LOCAL5`,
`LOG_LOCAL6`, `LOG_LOCAL7`, `LOG_USER`

firewall-soft-timeout

Timeout, in seconds, before the execution of a firewall control program is considered failed and a `SIGTERM` is sent to shutdown the process. Default is 15 seconds.

firewall-hard-timeout

Timeout, in seconds, before the shutdown of a firewall control program is considered failed and a `SIGKILL` is sent to the process. Default is 5 seconds.

backlog Specifies how many pending requests should be put in the queue of incoming requests before a "Connection refused" error message is sent to the client. Default is 50.

To define a new domain, use the following syntax:

```
domain NAME { DIRECTIVES }
```

`NAME` is used in the `DOMAIN` keyword sent in the request.

Recognized `DIRECTIVES` in a domain definition are:

max-users

Defines the maximum number of users allowed to be logged in concurrently. If the value specified is `-1`, no limit is enforced, which also is the default.

max-same-users

The maximum number a user can connect to the same account from different machines (different `MAC` addresses). Default is 1. The special value `-1` means no limit (except the `max-users` limit above).

firewall-program

The program to run when controlling the firewall. The firewall control program is called with the following parameters:

To open up a rule: `<open> <IP> <MAC>`

To close a rule: `<close> <IP> <MAC>`

To reset the firewall: `<reset>`

`IP` is a string in dotted-decimal form (`xxx.xxx.xxx.xxx`) and `MAC` is a colon-separated string of hex numbers (`00:01:60:12:a7:bd`).

This directive is mutually exclusive with the `fwcd-host` directive.

fwcd-host	Instead of running a local script to control the firewall, connect to this host (which must be running <code>fwcd</code> , the firewall control daemon).
fwcd-port	Remote port used when connecting to the firewall control daemon (using the <code>firewall-host</code> option above). The default port is 1717.
port	This is the port used to listen for connection from <code>fwcd</code> , when <code>fwcd</code> decides to logout a user.
certificate-file	This is the certificate used when identifying to <code>fwcd</code> . Default is <code>‘/usr/local/etc/oasis.cert’</code> .
key-file	The private key used when identifying to <code>fwcd</code> . Default is <code>‘/usr/local/etc/oasis.key’</code> .
fwcd-certificate-file	This is the certificate that will be expected from <code>fwcd</code> . Default is <code>‘/usr/local/etc/fwcd.cert’</code> .
pam-service	What PAM service to use to authenticate user on this domain. The PAM service name is also the name of the file in <code>‘/etc/pam.d/’</code> directory. This directive is mandatory.
probe-library	The plugin library to use to probe users. If no library or an empty string is specified, no probing will be performed.
probe-interface	Specifies what interface to use to probe users. The ARP probe plugin uses this when constructing the ARP packet.

2.4.5 Starting Oasis

Options are specified by the usual GNU command line syntax, with long options starting with two dashes (`--`).

Usage: `oasis [options]`

Options:

`‘-f’`

`‘--fg’` Run in foreground, don’t fork. When in foreground the log messages are sent directly to the console instead of to `syslog`.

`‘-r ‘FILE’’`

`‘--rcfile=‘FILE’’`

Read configuration file `‘FILE’` instead of the default one in `‘/usr/local/etc/oasis.conf’`.

`‘-d’`

`‘--debug’` Show debug messages.

‘-F’
 ‘--force’ Continue even if the socket already exists (it will be overwritten). Make sure no other oasis process is running.
 ‘-h’
 ‘--help’ Show a short help message describing the syntax.
 ‘-v’
 ‘--version’ Show the version number.

2.4.6 PAM

A complete description of PAM (Pluggable Authentication Modules) is outside the scope of this document. More information can be found at <http://www.kernel.org/pub/linux/libs/pam/>.

PAM is used to do the actual authentication of a user. PAM is a general framework for authentication, accounting and password management. Only the authentication part is used in the access server. The framework uses a plug-in system where modules can be written to extend the system to support almost any kind of authentication mechanism. Today there exists modules for Kerberos, Radius, LDAP, password files and more.

2.4.7 The firewall control program

If your domain is configured with `firewall-program`, this option specifies a program (a simple shell script) to be executed when controlling the firewall. In the scripts subdirectory of the source distribution there is a template to begin with.

The script is passed different parameters depending on what it should do. The first parameter is the action string which can be “open”, “close” or “reset”. The two first actions takes two more parameters, the IP and MAC address, as strings.

2.5 The firewall control daemon

`fwcd` is a daemon that runs on the firewall. This requires that the firewall is an ordinary computer and not a dedicated router (such as a Cisco router). It communicates with Oasis, who sends requests to `fwcd` to login a user. The firewall control daemon will then start to listen on the traffic to decide if the user is logged in or not. When the user is inactive (but still connected), `fwcd` starts to ping the user until he is active again.

This solves the problem of too much probing, since the firewall doesn’t need to actively probe all the time.

2.5.1 Starting `fwcd`

Options are specified by the usual GNU command line syntax, with long options starting with two dashes (--).

Usage: `fwcd [options]`

Options:

```

'-f'
'--fg'      Run in foreground, don't fork. When in foreground the log messages are sent
            directly to the console instead of to syslog.

'-r 'FILE''
'--rcfile='FILE''
            Read configuration file 'FILE' instead of the default one in
            '/usr/local/etc/fwcd.conf'.

'-d'
'--debug'   Show debug messages.

'-h'
'--help'    Show a short help message describing the syntax.

'-V'
'--version' Show the version number.

```

2.5.2 Configuring fwcd

The configuration file is by default installed as `‘/usr/local/etc/fwcd.conf’`. There is a template in the `‘etc’` subdirectory of the source distribution that you can modify to your needs. The syntax is simple: each directive consists of a keyword, an equal sign and an argument, as in:

```
keyword = argument
```

Comments begin with `#` or `//` and extends to the end of the line. Block comments can be written in C syntax, beginning with `/*` and ending in `*/`.

Valid directives are:

`probe-interval`

Time between consecutive checks of the users' traffic.

`log-facility`

What facility to use when logging to syslog. The following arguments are recognised:

```
LOG_AUTHPRIV, LOG_DAEMON, LOG_LOCAL0, LOG_LOCAL1,
LOG_LOCAL2, LOG_LOCAL3, LOG_LOCAL4, LOG_LOCAL5,
LOG_LOCAL6, LOG_LOCAL7, LOG_USER
```

`max-missed-probes`

Maximum number of probes a user can miss before the user is considered away and logged out by the system.

`firewall-soft-timeout`

Timeout, in seconds, before the execution of the firewall program is considered failed and a `SIGTERM` is sent to shutdown the process.

`firewall-hard-timeout`

Timeout, in seconds, before the shutdown of the firewall program is considered failed and a `SIGKILL` is sent to the process.

firewall-program

The program to run when controlling the firewall. The firewall program is called with the following parameters:

To open up a rule: `<open> <IP> <MAC>`

To close a rule: `<close> <IP> <MAC>`

To reset the firewall: `<reset>`

IP is a string in dotted-decimal form (xxx.xxx.xxx.xxx) and MAC is a colon-separated string of hex numbers (00:01:60:12:a7:bd).

port What port to use when listening for the connection from oasis.

probe-interface

What interface to use when listening for traffic and probing users.

oasis-host

The host oasis is running on.

oasis-port

The port oasis listens on for the connection from fwcd.

probe-library

The plugin library to use to probe users. If no library or an empty string is specified, no probing will be performed.

certificate-file

This is the certificate used when identifying to fwcd. Default is `'/usr/local/etc/fwcd.cert'`.

key-file The private key used when identifying to fwcd. Default is `'/usr/local/etc/fwcd.key'`.

oasis-certificate-file

This is the certificate that will be expected from oasis. Default is `'/usr/local/etc/oasis.cert'`.

network This is the network and mask we're listening on. Can be specified either on the format "192.168.1.0/24" or "192.168.1.0 mask 255.255.255.0".

promiscuous

Integer specifying whether to sniff packets in promiscuous mode or not. Generally, setting **promiscuous** to 0 is a better choice since the firewall only needs to listen on traffic to be routed through that machine.

3 Step-by-step installation instructions

This is a step-by-step guide for installing the StockholmOpen.Net system. It is assumed that the reader has knowledge of Un*x administration in general and Un*x networking in particular. Installation of the StockholmOpen.Net system is more of "plug-and-pray" than "plug-and-play" right now.

1. Choose machines

Choose what machines should run the configuration services (BOOTP relay agent, MAC database and registration). Choose if the access server and firewall should be colocated in the same machine. A tip is to start with one machine for access server and firewall if you're not sure you will need them separated (perhaps you're using a Cisco router?). They can easily be separated later.

2. Setup the common configuration services:

On the configuration machine, install a PostgreSQL database and run the initialization script (with your modifications) according to the instructions at Section 2.3 [Database], page 5.

Download and unpack the DHCP 3.0 sources from ISC. Patch and install the relay agent according to the instructions in Section 2.1 [BOOTP relay agent], page 3.

The relay agent machine needs one additional interface for each operator present in order for the DHCP server to accept the request as authoritative. These interfaces should be aliases and assigned an IP address belonging to the operators subnet. In Linux aliased interfaces can be created with `ifconfig eth0:0 10.0.0.1` (the first alias, starting with 0, for interface eth0).

Install a webserver (eg Apache) with PHP support and the relay web pages. Edit the settings in `config.php`. PHP should be compiled with PostgreSQL support.

3. For each upstream provider:

Download and install the oasis access server. Configure and build it according to Section 2.4.3 [Compiling Oasis], page 7. Configure oasis: Section 2.4.4 [Oasis Configuration], page 7.

Edit the firewall script. By default it is located in `/usr/local/sbin/oasis-firewall-sample`. If the firewall is colocated with the access server (ie, local access to the firewall), the sample script will probably work fine (assuming you use iptables), otherwise use your imagination (SSH could be used to securely control a remote PC-based firewall, in which case you must set up public keys appropriately for automatic root access).

In the reset section you might want to include static rules for access to an upstream DNS and other services that should be provided before authentication.

The default firewall script includes redirection to a local webserver for unauthenticated users. If you decided to separate oasis and the firewall this will not work directly. Install a simple webserver on the firewall that issues a http redirect to the oasis web server.

Start oasis and test it with the included `client` program. It's available in the `src` subdirectory in the source distribution.

Download and install an SSL secured web server (eg Apache-SSL) with PHP support. Use the PHP web pages from oasis (by default installed in `/usr/local/var/www`) and tailor it to your taste. They are quite minimal in order to make it easier to modify the design/layout.

Configure the authentication backend. PAM is used for this and by default the PAM service in the oasis configuration file is set to “login”, ie local UNIX authentication (using `/etc/passwd`). You might want to use some other authentication mechanism such as Kerberos or RADIUS. Install the PAM module (usually under `/lib/security`) and create a PAM configuration file for oasis.

Assuming you name your PAM service “gazonk”, create the file `/etc/pam.d/gazonk` with the following contents for a Kerberos service (a Kerberos 4 PAM module is available at <http://software.stockholmopen.net/>):

```
auth required pam_krb4.so MY.REALM
```

Then point oasis to this PAM service by setting `pam-service` in the oasis configuration file to “gazonk”. Now oasis will use the PAM rules in `/etc/pam.d/gazonk` when authenticating users for that domain. You can have different authentication mechanisms for different domains.

PAM is very flexible. For example to use an existing Kerberos database but only allow a subset of the users in the Kerberos database, one can use the following in the PAM configuration file:

```
auth required pam_listfile.so onerr=fail sense=allow item=user file=/etc/userlist
auth required pam_krb4.so
```

Make sure there is an entry in the PostgreSQL database (in the `T_provider` table) for this provider.