Dante, Module *Redirect*

Inferno Nettverk A/S
Oslo Research Park
Gaustadalléen 21
NO-0349 Oslo
Norway

Date: 2001/11/13 09:16:12

1 Description

The *Redirect* module gives you control over both where clients requests and replies will end up, and what addresses and portranges the Dante server will use on behalf of the clients for outgoing connections.

It can be used to redirect clients connections from one address to another, useful for cases where you for instance want clients to go via a local web-proxy instead of directly to the webserver.

It can be used to restrict the portranges used by the Dante server, useful for cases where a firewall needs to know what portranges the Dante server will use.

A stranger but also possible use is to limit the number of sessions clients can establish.

2 Syntax

The syntax of the `redirect` statement is as follows:

`redirect from: <address> to: <address>`

Either `from` or `to` are optional, but at least one must be given.

`address` are addresses in the usual Dante format.

3 Semantics

The `redirect` statement integrates as a part of socks-rules.

The meaning of `to` and `from` varies considerably depending on what socks `command` the `redirect` statement applies to. The next section will detail the semantics of each *redirect* application based on the `command` used (with the corresponding `protocol` in parenthesis, for those preferring that).

3.1 `bind` (`protocol: tcp`)

`from` is the address to bind on behalf of the client.
`to` is ignored.

3.2 `bindreply` (`protocol: tcp`)

`from` is the address to tell the client the bindreply (connection) is from.
`to` is the address to send the bindreply (connection) to.

3.3 `connect` (`protocol: tcp`)

`from` is the address to use on behalf of the client for making the connection.
`to` is the address to connect the client to.

3.4 `udpassociate` (`protocol: udp`)

`from` is the address to use on behalf of the client for sending udp packets.
`to` is the address to send packets from the client to.

3.5 `udpreply` (`protocol: udp`)

`from` is the address to tell the client the reply is from.
`to` is the address to send the reply to.

4 Examples

This section shows several examples of how one could use the *redirect* module.

4.1 Redirecting web-requests to a web proxy

The below rule redirects clients from the `10.0.0.0/24` net who want to connect to the http port of any address to the address `squid.example.com`, port `3128`.

```
pass {
   from: 10.0.0.0/24 to: 0.0.0.0/0 port = http
   command: connect
   redirect to: squid.example.com port = 3128
}
```

4.2 Limiting the portranges used by the *Dante* server

The next rule says that the server should limit itself to using portranges above `32768` on the interface `de1` when sending packets out on behalf of the clients on the `10.1.1.0/24` net.

```
pass {
   from: 10.0.0.0/24
   redirect from: de1 port > 32768}
}
```

4.3 Limiting concurrent user sessions

A more novel use of the `redirect` statement allows one, in a rather ad-hoc way, to limit the number of concurrent user sessions.

The next rule says the server should limit itself to using the portrange `2048 - 2051` on the interface `de1`, a range of four portnumbers, for the client 10.0.0.1.

```
pass {
   from: 10.0.0.1/32
   redirect from: de1 port 2048 - 2051
}
```

When the client 10.1.1.1 reaches four concurrent sessions, it will not be able to create new ones until it closes one of the previously established sessions.

For this to work correctly it is however required that no other client accidently ends up using the same portrange. One way to accomplish this is to make the interfacename `de1` be used uniquely for this sort of thing, or to make sure all other rules use a different portrange by way of having their own `redirect` statement. The former is greatly preferred.

As can probably be guessed, this is a new usage of the `redirect` statement that was not intended in the design, but was discovered by accident and is mentioned here since it could be useful.