Dante Module Documentation
Redirect Module

Inferno Nettverk A/S

Date: 2021/04/28 17:36:32

1 Description

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

The module can be used to redirect client connections from one address to another, which can be useful in cases where clients should use a local web-proxy instead of communicating directly with external web servers.

It can also be used to restrict the port ranges used by the *Dante* server, which can be useful in cases where a firewall needs to know which port ranges the *Dante* SOCKS server will use.

Additionally, it can be used to make the IP-address the *Dante* server will use when connecting to a remote server be chosen based on the IP-address of each connecting client, or even the username, if username-based authentication is configured.

2 Syntax

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

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

It is not necessary to specify both, but at least one of the `from` or `to` keywords is needed.

Here `address` can be an address in any format supported by *Dante*. See *sockd.conf(5)* for more information about this.

If the `to` address does not contain a port-specifier, the *Dante* server will use the same port as the original socks-request, where applicable. This also makes it possible to redirect all connections for one host to another, without having to specify one redirect statement for each port number.

3 Semantics

The `redirect` statement can be used in both *client-rules* or *socks-rules*. See *sockd.conf(5)* for more information about the different rule types.

Note that a redirection set in a client-rule *is not necessarily inherited by a later* socks-rule. *Whether it is or not depends on the command used in the* socks-rule.

For some commands, such as *connect*, inheritance makes sense, while for others, such as *bind*, this does not make sense, and there is no inheritance applied by the *Dante* server. The section listing the semantics of each *redirect* application lists the commands for which a redirection setting is inherited.

The intent is that redirection will be inherited where it makes sense.

The meaning of `to` and `from` varies depending on which SOCKS `command` the `redirect` statement applies to.

The next section details the semantics of *redirect*, based on the `command` used (with the corresponding `protocol` in parenthesis).

3.1 `bind (protocol: tcp)`

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

Redirection is not inherited from client-rule*s.*

3.2 `bindreply (protocol: tcp)`

`from` *is the address the client is told the bindreply connection is from.*
`to` *is the address to send the bindreply connection to (only applicable if using the bind extension).*
Redirection is not inherited from client-rule*s.*

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.*
The redirection from *address is inherited from* client-rule*s.*

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.*
The redirection from *address is inherited from* client-rule*s.*

3.5 `udpreply (protocol: udp)`

`from` *is the address to tell the client the reply is coming from.*
`to` *is the address to send the reply to.*
Redirection is not inherited from client-rule*s.*

4 Examples

This section shows several examples of how the redirect *module can be used.*

4.1 Redirecting web-requests to a web proxy

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

```
socks 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 port ranges used by the *Dante* server

The next rule makes the server limit itself to using ports above `32768` on the interface `de1` when sending out packets on behalf of the clients on the `10.1.1.0/24` network.

```
socks pass {
   from: 10.0.0.0/24 to: 0.0.0.0/0
   redirect from: de1 port > 32768
}
```

4.3 Using different IP-addresses for different clients

The next two rules show how the Dante server could be instructed to use different IP-addresses for different clients.

```
# clients from the 10.1/16-net will be assigned IP-address 192.168.0.1
socks pass {
   from: 10.1.0.0/16 to: 0.0.0.0/0
   redirect from: 192.168.0.1/32
}

# clients from the 10.2/16-net will be assigned IP-address 192.168.0.2
socks pass {
   from: 10.2.0.0/16 to: 0.0.0.0/0
   redirect from: 192.168.0.2/32
}
```