Inferno Nettverk A/S

Dante Module Documentation

Bandwidth Module

January 8, 2017

1 Description

The *Bandwidth* module gives control over how much bandwidth the clients of the *Dante* SOCKS server can consume.

The module can be used to limit bandwidth to non-work related web/FTP sites, or to prevent FTP-related traffic from impacting too much on interactive telnet/ssh traffic.

It can also be used to give more bandwidth to certain clients or for traffic to certain sites.

When combined with the *Dante bind extension*, the module can be used to provide bandwidth control for network servers (like e.g., web servers) that do not have support for bandwidth control.

2 Syntax

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

`bandwidth: <bytes>`

`bytes` is the maximum bandwidth to use per second, measured in bytes.

3 Semantics

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

Note that a bandwidth limitation set in a client-rule *is inherited by the* socks-rule *matching the client.*

The maximum allowed `bandwidth` set for a rule will be shared by all clients matching that rule. The *Dante* server will attempt to distribute the bandwidth to the matching clients in a least-recently used fashion, trying to let all clients get a fair share.

3.1 Special notes

Note that for UDP, as for TCP, the setting is based on the rule matching the TCP-based control-connection, not on each individual UDP packet.

A full ACL-check is done for each UDP packet, but the limits are enforced *based on the rule matched for the control-connection only*.

4 Examples

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

4.1 Limiting web/http bandwidth

The below rule shows how to limit the bandwidth used for web traffic for the clients on the 10.0.0.0/24 network to a total of 10240 bytes (10 KiloBytes/second).

```
client pass {
   from: 10.0.0.0/24 to: 0.0.0.0/0 port = http
   command: connect
   bandwidth: 102400
```

```
}
```

4.2 Increasing web/http bandwidth

The next rule, if placed before other bandwidth-limiting rules, shows how one can increase the bandwidth used for web traffic by the clients on the 10.0.0.0/24 network to a specified host.

In this case, the clients will be able to use 1024000 bytes (one MegaByte/second), when getting data from the host *work.example.com*.

```
socks pass {
   from: 10.0.0.0/24 to: work.example.com port = http
   command: connect
   bandwidth: 1024000
}
```

4.3 Limiting FTP bandwidth

The next rule shows how one can limit the bandwidth used for FTP data transfers for the clients on the 10.0.0.0/24 network to a total of 10240 bytes (10 KiloBytes/second).

This only works for *active* FTP, since for *passive* FTP there are no fixed port numbers.

```
socks pass {
   from: 0.0.0.0/0 port = ftp-data to: 10.0.0.0/24
   command: bindreply
   bandwidth: 10240
}
```

4.4 Limiting bandwidth provided by internal servers to the outside

The next rule shows how one could use the *Dante bind extension* together with the *Bandwidth* module to limit the amount of data provided by an internal server, in this case, a web server called *our-webserver.example.com*, to a total of 10240 bytes, or 10 KiloBytes/second.

This requires the webserver to be socksified and the *bind extension* to be enabled on both the socksified client and on the *Dante* server.

```
socks pass {
   from: 0.0.0.0/0 to: our-webserver.example.com port = http
   command: bindreply
   bandwidth: 10240
}
```

5 Special notes

5.1 SIGHUP

Sending the server a `SIGHUP` signal forces a reload of the configuration file. It should be noted that *this does not* affect current sessions or limits placed on them.

A reload of the configuration file only affects sessions created after the reload. It will not affect any of the existing sessions.

This means that changing e.g., a *pass* statement to a *block* statement, does not terminate the session of any existing client. Likewise, changing the limits set in a rule does not change the values for any existing session.

After a reload of the configuration file, old sessions will continue to operate in a separate space, using the old configuration, while new sessions will use the new configuration.