
Section 6 -- NetNAT Commands

The NetNAT supports a variety of commands that allow you to customize the NAT configuration and operational characteristics to meet the specific requirements of your network. NetNAT configuration commands fall into the following categories:

| | |
|------------------------|---|
| Global | For defining NAT operational characteristics |
| Interface | For creating, configuring, or deleting an interface |
| Service Mapping | For controlling address translation |
| Syslog Service | For managing remote logging activities |
| IP Routing | For defining specific routes for NetNAT to use when sending messages. |

Command Description

The description for each command contains the following information:

| | |
|---------------------------|---|
| Command Name | A string value that uniquely identifies the command. The command name appears as the heading for the command description. When typing the command name, you must enter the characters exactly as shown. |
| Purpose | Identifies the task this command performs. |
| Command Syntax | Indicates the format to use when typing the command and its associated parameters. |
| Command Parameters | Required or Optional pieces of information that you must supply to set options for command. |
| For Example | Provides an example of how the command syntax appears when all the parameters are given real values. |
| Description | When appropriate, provides additional information about the command. |

Command Abbreviation

The commands may be abbreviated on input to the shortest representation that is still unique. For abbreviations that are not unique, the command earlier in the search sequence is used. The order that commands exist in the search sequence may be inferred by the order they are displayed in the help command output.

Global Commands

The *Global* commands listed below allow you to set the operational characteristics for the NAT and start and stop the NetNAT servers.

| | |
|----------|--|
| hostname | Sets the hostname for the NAT. |
| auth | Starts the BSD Authentication server |
| discard | Starts the discard server |
| echo | Starts the echo server |
| www | Starts the Web server |
| watchdog | Sets the timeout period for the Watchdog Timer |
| exit | Shuts down the NAT and returns to DOS. |

Interface Commands

The *Interface* commands listed below allow you attach drivers and configure interfaces between the NAT and the physical network(s). These commands must be completed before the NAT can communicate on the network interfaces.

| | |
|---------------|--|
| attach alias | Creates a new name for an existing NAT logical interface. |
| attach asy | Allows you to attach an asynchronous hardware driver to the NAT logical interface. |
| attach packet | Allows you to attach an ethernet hardware driver to a NAT logical interface. |
| ifconfig | Sets the parameters between the attached hardware drivers and the NAT logical interface. Also displays the status for the NAT logical interface. |

Interface Naming Conventions

While this is not a major issue, we encourage the use of predictable interface names for particular interface types. The number of available interfaces depends on the NAT model you are using. Use the table below to select names for your interfaces:

| Interface Type | Recommended Name |
|---------------------|--------------------|
| Ethernet | en0, en1, en2, en3 |
| Alias for Ethernet | en0a, en0b, en0c |
| Token Ring | tr0, tr1 |
| PPP Interfaces | ppp0, ppp1 |
| VLAN Trunks | trk0, trk1 |
| VLAN Sub-Interfaces | sub0, sub1 |

Service Mapping Commands

The *Service Mapping* commands listed below instruct the NetNAT to create new mapping relationships between apparent external services and actual internal services. The mapping relationships may be port-and-protocol conscious, or port independent.

| | |
|-----------------|---|
| Port Mapping | Allows you to map a specific apparent IP Address/Port combination to an internal actual service. |
| Fixed Mapping | Allows you to map all services for an external apparent IP Address to an internal IP Address. |
| Default Mapping | Allows you to map all undefined services to an internal IP Address. |
| Local Mapping | Do not map this service, as it is processed locally by the NetNAT. |
| Mux Mapping | Distributes incoming service requests across an array of servers. |
| Pool Mapping | Allocates public IP Addresses temporarily to internal clients on a one-to-one basis. |
| Subnet Mapping | Assigns public IP Addresses based upon subnet rules. This also may enable SNMP payload translation. |

Route Commands

The *Route* commands listed below allow you to define routing information for the NetNAT to use when accessing the internet, specific external networks, and internal, private networks.

| | |
|-------------------|---|
| route add default | Allows you to set the default gateway for accessing the internet. |
| route add | Allows you to set a public route to a specific network. |
| route addprivate | Allows you to set a private route to a specific network. |
| route tunnel | Creates a VPN tunnel to a specific network. |

Syslog Service Commands

The Syslog commands listed below allow you to enable or disable the logging of NetNAT events and statistics and to turn on and off syslog's reject message logging facilities.

| | |
|---------------|--|
| syslog | Allows you to start or change the syslog server, display the current status of the syslog, and stop the syslog server. |
| syslog reject | Allows you to enable or disable the syslog reject message logging facilities. |

access

Purpose

access displays or modifies a set of rules that control the access of **external** networks by **internal** users. This facility permits the firewall administrator to limit direct Internet access to a subset of internal users if desired. Blocking internal users does not necessarily prevent those users from using the Internet, since they may be using a proxy server to access the Internet when that proxy server isn't blocked.

Command Syntax

```
access
access permit [subnet address] [netmask]
access deny [subnet address] [netmask] {log}
access drop [subnet address] [netmask]
```

Command Parameters

| Parameter Name | Description |
|----------------|--|
| permit | This specifies that the described subnet or host should be given unlimited Internet access rights. |
| deny | This specifies that the described subnet or host should be prevented from accessing the Internet. |
| drop | This specifies that a previously defined rule for the described subnet or host should be removed. |
| subnet address | The IP address of a subnet or host that we wish to limit or enable Internet access. |
| netmask | The netmask of the subnet or 255.255.255.255 if specifying a host. |
| log | This optional parameter on the access deny subcommand causes a rejected attempt to be logged via syslog, using the priority specified for all rejected connections. See the "syslog reject" command. |

For Example

To display the current access control:

```
nscnat> access
4 access control rules in effect:
  10.1.0.0/16 (255.255.255.0) Local only (3284 uses)
  10.1.1.0/24 (255.255.255.0) Unlimited (9916628 uses)
  10.1.3.6/32 (255.255.255.255) Unlimited (579932 uses)
  10.1.9.99/32 (255.255.255.255) Unlimited (10966 uses)
nscnat> _
```

This shows that the huge subnet 10.1.x.x is denied direct access to the Internet, while within that "denied" subnet, the Class-C subnet 10.1.1.x has unlimited access, and the

two hosts 10.1.3.6 (the proxy server and mail relay) and 10.1.9.99 (the network administrator) also have unlimited direct access.

Usage counts are displayed for each rule.

access permit

Purpose

access permit creates a rule that gives a subnet or host unlimited direct access to the Internet.

Command Syntax

```
access permit [subnet address] [netmask]
```

Command Parameters

| Parameter Name | Description |
|----------------|--|
| subnet address | The IP address of a subnet or host that we wish to give Internet access. |
| netmask | The netmask of the subnet or 255.255.255.255 if specifying a host. |

For Example

Here we display the current rules, add an unlimited user and redisplay the rules to see if the addition worked:

```
nscnat> access
4 access control rules in effect:
  10.1.0.0/16 (255.255.255.0) Local only (3284 uses)
  10.1.1.0/24 (255.255.255.0) Unlimited (9916628 uses)
  10.1.3.6/32 (255.255.255.255) Unlimited (579932 uses)
  10.1.9.99/32 (255.255.255.255) Unlimited (10966 uses)
nscnat> access permit 10.1.9.100 255.255.255.255
nscnat> access
5 access control rules in effect:
  10.1.0.0/16 (255.255.255.0) Local only (3284 uses)
  10.1.1.0/24 (255.255.255.0) Unlimited (9916628 uses)
  10.1.3.6/32 (255.255.255.255) Unlimited (579932 uses)
  10.1.9.99/32 (255.255.255.255) Unlimited (10966 uses)
  10.1.9.100/32 (255.255.255.255) Unlimited (0 uses)
nscnat> _
```

Additional Information

This command may be used to change a rule for a previously specified network. If a rule already exists for the network we wish to grant unlimited access, the restriction is changed to “unlimited” from whatever it was before.

access deny

Purpose

access deny creates a rule that denies a subnet or host unlimited direct access to the Internet.

Command Syntax

```
access deny [subnet address] [netmask] {log}
```

Command Parameters

| Parameter Name | Description |
|----------------|--|
| subnet address | The IP address of a subnet or host that we wish to deny direct Internet access. |
| netmask | The netmask of the subnet or 255.255.255.255 if specifying a host. |
| log | This optional parameter, when present, instructs the NetNAT to log denied accesses via the reject syslog facility. By default, the reject syslog messages use the LOCAL2 facility and the INFO priority. This may be modified by the "syslog reject" command. The logged messages will have a record ID of "ac" to indicate that these represent denied outbound accesses. |

For Example

Here we display the current rules, add a restricted user and redisplay the rules to see if the addition worked:

```
nscnat> access
4 access control rules in effect:
  10.1.0.0/16 (255.255.255.0) Local only (3284 uses)
  10.1.1.0/24 (255.255.255.0) Unlimited (9916628 uses)
  10.1.3.6/32 (255.255.255.255) Unlimited (579932 uses)
  10.1.9.99/32 (255.255.255.255) Unlimited (10966 uses)
nscnat> access deny 10.1.9.100 255.255.255.255
nscnat> access
5 access control rules in effect:
  10.1.0.0/16 (255.255.255.0) Local only (3284 uses)
  10.1.1.0/24 (255.255.255.0) Unlimited (9916628 uses)
  10.1.3.6/32 (255.255.255.255) Unlimited (579932 uses)
  10.1.9.99/32 (255.255.255.255) Unlimited (10966 uses)
  10.1.9.100/32 (255.255.255.255) Local only (0 uses)
nscnat> _
```

Additional Information

This command may be used to change a rule for a previously specified network. If a rule already exists for the network we wish to restrict to local access, the restriction is changed to "local only" from whatever it was before.

access ftp

Purpose

access ftp creates a rule that denies a subnet or host direct access to the Internet, except for FTP accesses. This is to accommodate applications that use both HTTP and FTP for data transfer, but that have added HTTP proxy support and not FTP proxy support.

Command Syntax

```
access ftp [subnet address] [netmask] {log}
```

Command Parameters

| Parameter Name | Description |
|----------------|--|
| subnet address | The IP address of a subnet or host that we wish to deny direct Internet access except for FTP transfers. |
| netmask | The netmask of the subnet or 255.255.255.255 if specifying a host. |
| log | This optional parameter, when present, instructs the NetNAT to log denied accesses via the reject syslog facility. By default, the reject syslog messages use the LOCAL2 facility and the INFO priority. This may be modified by the "syslog reject" command. The logged messages will have a record ID of "ac" to indicate that these represent denied outbound accesses. |

For Example

Here we display the current rules, add a restricted user and redisplay the rules to see if the addition worked:

```
nscnat> access
4 access control rules in effect:
  10.1.0.0/16 (255.255.255.0) Local only (3284 uses)
  10.1.1.0/24 (255.255.255.0) Unlimited (9916628 uses)
  10.1.3.6/32 (255.255.255.255) Unlimited (579932 uses)
  10.1.9.99/32 (255.255.255.255) Unlimited (10966 uses)
nscnat> access ftp 10.1.9.100 255.255.255.255
nscnat> access
5 access control rules in effect:
  10.1.0.0/16 (255.255.255.0) Local only (3284 uses)
  10.1.1.0/24 (255.255.255.0) Unlimited (9916628 uses)
  10.1.3.6/32 (255.255.255.255) Unlimited (579932 uses)
  10.1.9.99/32 (255.255.255.255) Unlimited (10966 uses)
  10.1.9.100/32 (255.255.255.255) FTP (0 uses)
nscnat> _
```

Additional Information

This command may be used to change a rule for a previously specified network. If a rule already exists for the network we wish to restrict to local access, the restriction is changed to "FTP" from whatever it was before.

access drop

Purpose

access drop deletes a rule that was created with either an access permit or access deny command.

Command Syntax

```
access drop [subnet address] [netmask]
```

Command Parameters

| Parameter Name | Description |
|----------------|--|
| subnet address | The IP address of a subnet or host that we specified in a previous permit or deny command. |
| netmask | The netmask of the subnet or 255.255.255.255 if we specified a host. |

For Example

Here we display the current rules, drop an unlimited user and redisplay the rules to see if the deletion worked:

```
nscnat> access
4 access control rules in effect:
  10.1.0.0/16 (255.255.255.0) Local only (3284 uses)
  10.1.1.0/24 (255.255.255.0) Unlimited (9916628 uses)
  10.1.3.6/32 (255.255.255.255) Unlimited (579932 uses)
  10.1.9.99/32 (255.255.255.255) Unlimited (10966 uses)
nscnat> access drop 10.1.3.6 255.255.255.255
nscnat> access
3 access control rules in effect:
  10.1.0.0/16 (255.255.255.0) Local only (3284 uses)
  10.1.1.0/24 (255.255.255.0) Unlimited (9916679 uses)
  10.1.9.99/32 (255.255.255.255) Unlimited (10968 uses)
nscnat> _
```

access rst

Purpose

access rst controls the transmission of the TCP RST packet to an internal user when an access attempt is being rejected. The normal TCP tradition calls for a failed attempt to be rejected with a reset packet (containing the RST flag). This will, however, result in an immediate access retry by many programs. For example, Microsoft Internet Explorer will retry four times for each configured MSN host, and Netscape Navigator will retry nearly 100 times before giving up. If you are logging the rejected attempts, this can get pretty upsetting after a while. Thus, we have created the access rst command to permit you to disable the RST messages.

Command Syntax

```
access rst [on|off]
```

Command Parameters

| Parameter Name | Description |
|----------------------|---|
| on true yes 1 | Enable transmission of the RST packet in response to rejected outbound TCP connection. |
| off false no 0 | Disable transmission of the RST packet in response to rejected outbound TCP connection. |

Additional Information

The access rst command without the on/off parameter will display the current state of the access rst mode.

arp

Purpose

arp displays the contents of the IP Address-to-ethernet address translation table, and through sub-commands allows you to manipulate that table. This section describes the arp command in its most basic form, and lists the sub-commands. Subsequent sections will detail the sub-command functions.

Command Syntax

```
arp
arp add [hostid] [ether addr]
arp drop [hostid]
arp flush
arp publish [hostid] [ether addr]
```

Command Parameters

| Parameter Name | Description |
|----------------|---|
| hostid | The hostname or IP Address of the specified computer. |
| ether addr | The hardware or MAC address for the ethernet card. This is specified as six two-digit hex values separated by colons. |

For Example

To display the contents of the ARP table:

```
nscnat> arp
received 2 badtype 0 bogus addr 0 reqst in 0 replies 2 reqst out 2
IP addr      Type      Time Q Addr
139.31.113.12 10 Mb Ethernet 899 00:20:af:eb:01:de
139.31.113.41 10 Mb Ethernet 712 00:60:be:83:23:11
nscnat> _
```

On the first line of displayed output:

| | |
|------------|---|
| received | Number of unknown hardware types in received messages |
| badtype | Number of arp messages received. |
| bogus addr | Number of times computers claimed invalid IP Addresses. |
| reqst in | Number of received ARP requests. |
| reqst out | Number of transmitted ARP requests. |

In the ARP Table Display section:

| | |
|---------|--|
| IP addr | Address of a computer on the LAN. |
| Type | Hardware type (ethernet only). |
| Time | Number of seconds until this entry expires. |
| Q | If the ARP process is still pending, the number of messages in queue awaiting address resolution before they may be sent. |
| Addr | The hardware or "MAC" address of the remote computer's ethernet controller, expressed as six two-digit hex values separated by colons. |

Additional Information

The sub-commands that manipulate the ARP table are described in the next sections.

arp add

Purpose

The arp add sub-command allows you to create an ARP table entry by hand. Normally, this process occurs automatically in the course of normal network operation, but you may wish to manually define a host's hardware address. Entries created with the arp add sub-command do not expire as do normal entries. The only way to delete these is with the arp drop command.

Command Syntax

```
arp add [hostid] [ether addr]
```

Command Parameters

| Parameter Name | Description |
|----------------|---|
| hostid | The hostname or IP Address of the specified computer. |
| ether addr | The hardware or MAC address for the ethernet card. This is specified as six two-digit hex values separated by colons. |

For Example

Here we display the ARP table, manually add an entry, and then re-display the table to show that the add worked:

```
nscnat> arp
received 2 badtype 0 bogus addr 0 reqst in 0 replies 2 reqst out 2
IP addr      Type      Time Q Addr
139.31.113.12 10 Mb Ethernet 899 00:20:af:eb:01:de
139.31.113.41 10 Mb Ethernet 712 00:60:be:83:23:11
nscnat> arp add 139.31.113.69 00:20:af:eb:02:ea
nscnat> arp
received 2 badtype 0 bogus addr 0 reqst in 0 replies 2 reqst out 2
IP addr      Type      Time Q Addr
139.31.113.12 10 Mb Ethernet 899 00:20:af:eb:01:de
139.31.113.41 10 Mb Ethernet 712 00:60:be:83:23:11
139.31.113.69 10 Mb Ethernet 0    00:20:af:eb:02:ea
nscnat> _
```

arp drop

Purpose

The arp drop sub-command will delete a specified ARP table entry by hand. Normally, this process occurs periodically in the course of normal network operation, but you may wish to do this for debugging reasons.

Command Syntax

```
arp drop [hostid]
```

Command Parameters

| Parameter Name | Description |
|----------------|---|
| hostid | The hostname or IP Address of the specified computer. |

For Example

Here we display the ARP table, manually drop an entry, and then re-display the table to show that the drop worked:

```
nscnat> arp
received 2 badtype 0 bogus addr 0 reqst in 0 replies 2 reqst out 2
IP addr      Type      Time Q Addr
139.31.113.12 10 Mb Ethernet 899 00:20:af:eb:01:de
139.31.113.41 10 Mb Ethernet 712 00:60:be:83:23:11
139.31.113.69 10 Mb Ethernet 0   00:20:af:eb:02:ea
nscnat> arp drop 139.31.113.69
nscnat> arp
received 2 badtype 0 bogus addr 0 reqst in 0 replies 2 reqst out 2
IP addr      Type      Time Q Addr
139.31.113.12 10 Mb Ethernet 899 00:20:af:eb:01:de
139.31.113.41 10 Mb Ethernet 712 00:60:be:83:23:11
nscnat> _
```

If the entry that you delete is for an active host, it will be restored almost before you can look back.

arp flush

Purpose

The arp flush sub-command completely clears the ARP table. This is a debugging tool that may help occasionally. Active hosts will return immediately to the table, as the ARP process re-resolves their hardware addresses. Please note that table entries that were created by hand using the arp add facility will not be flushed by this sub-command.

Command Syntax

arp flush

Command Parameters

None.

For Example

Here we display the ARP table, flush the table, and then re-display it:

```
nscnat> arp
received 2 badtype 0 bogus addr 0 reqst in 0 replies 2 reqst out 2
IP addr      Type      Time Q Addr
139.31.113.12 10 Mb Ethernet 899 00:20:af:eb:01:de
139.31.113.41 10 Mb Ethernet 712 00:60:be:83:23:11
139.31.113.69 10 Mb Ethernet 0   00:20:af:eb:02:ea
nscnat> arp flush
nscnat> arp
received 2 badtype 0 bogus addr 0 reqst in 0 replies 2 reqst out 2
IP addr      Type      Time Q Addr
139.31.113.69 10 Mb Ethernet 0   00:20:af:eb:02:ea
nscnat> _
```

If the entry that you delete is for an active host, it will be restored almost before you can look back. If the entry was created by (as was the persistent entry in this example), it will not be deleted by the arp flush sub-command.

arp publish

Purpose

The `arp publish` sub-command is a special-case version of `arp add`, which, in addition to creating an ARP table entry for the specified host, instructs the NetNAT to reply to ARP requests for that host. This may be used to help a host that seems incapable of replying to ARP requests, or when you wish the NetNAT to publish its own hardware address when the NetNAT can forward packets addressed for the specified host to its destination. When the NetNAT is answering ARP requests on behalf of a host that the NetNAT can help reach, it is called “proxy ARP.” As in `arp add`, the entries created by `arp publish` do not expire.

Command Syntax

```
arp publish [hostid] [ether addr]
```

Command Parameters

| Parameter Name | Description |
|----------------|---|
| hostid | The hostname or IP Address of the specified computer. |
| ether addr | The hardware or MAC address for the ethernet card. This is specified as six two-digit hex values separated by colons. |

This command is infrequently used, but can be very helpful when routing protocols fail.

attach packet

Purpose

`attach packet` is an *interface* command that creates the logical interface between a hardware driver and the NetNAT. Refer to *Interface Naming Conventions* earlier in this chapter for recommended interface names.

Command Syntax

```
attach packet [int #] [int name] [pool size] [mtu]
```

Command Parameters

| Parameter Name | Description |
|----------------|--|
| int # | The software interrupt vector assigned to the hardware driver when it was started. |
| int name | The name of the NetNAT's logical interface. |
| pool size | The size in K-bytes of a memory pool to allocate for the low-level driver software to use. |
| mtu | The physical size in bytes of maximum packet that this interface will be able to send. |

For Example

To attach the first (or only) driver to NAT interface "en0", the command is:

```
attach packet 0x60 en0 32 1500
```

This command attaches a packet driver on software interrupt vector 0x60 (96 decimal) to interface "en0" and allocates 32 K-Bytes of buffering for 1500-byte packets.

Possible responses are:

- ◆ "Interface xxx already exists" when the specified interface already identifies an attached hardware driver.
- ◆ "No packet driver loaded at int 0xnn" when there is no hardware driver installed on the specified software interrupt number.
- ◆ No response when the command is successful.

attach alias

Purpose

`attach alias` is an *interface* command that associates a new name with an existing network interface. This permits assignment of additional IP Addresses, service mappings and routes to a single ethernet interface.

Command Syntax

```
attach alias [int name] [alias name]
```

Command Parameters

| Parameter Name | Description |
|----------------|--|
| int name | The name of an existing NetNAT ethernet interface. |
| alias name | The new name to be associated with the existing interface. |

For Example

To create a new alias "en0a" for NAT interface "en0", the command is:

```
attach alias en0 en0a
```

The resulting alias interface inherits the primary characteristics of the true interface, including IP Address and hardware characteristics.

Possible responses are:

- ◆ "Main interface xxx not found" when the specified network interface does not exist.
- ◆ "Interface xxx already exists" when the specified alias interface already exists as either an actual or alias interface.
- ◆ "Interface xxx is an alias. May not alias an alias." when the specified network interface is itself an alias of another actual network interface.
- ◆ "Interface xxx not an ethernet." when the specified network interface is not an ethernet interface.
- ◆ No response when the command is successful.

attach vlan

Purpose

`attach vlan` is an *interface* command that creates a logical sub-interface within a VLAN Trunk. The VLAN Trunk must be compatible with the IEEE 802.1Q Tag Switching specification. Refer to *Interface Naming Conventions* earlier in this chapter for recommended interface names.

Command Syntax

```
attach vlan [trunk int name] [sub-int name] [vlan label]
```

Command Parameters

| Parameter Name | Description |
|----------------|---|
| trunk int name | The name of a previous-defined ethernet interface that was designated as a VLAN Trunk through use of the <code>ifconfig vlan</code> command. |
| sub-int name | The name of the new logical interface that is a sub-interface within the specified VLAN Trunk. |
| vlan label | The 20-bit decimal label specifying the VLAN within this trunk. See the 802.1Q specification and the documents that accompany 802.1Q-compliant ethernet switches. |

For Example

To attach the first (or only) driver to NAT interface "sub0" within trunk "trk0" on label 11202, the command is:

```
attach vlan trk0 sub0 11202
```

Possible responses are:

- ◆ "Interface xxx already exists" when the specified interface already identifies an attached hardware driver.
- ◆ No response when the command is successful.

attach asy

Purpose

attach asy is an *interface* command that creates the logical interface between an asynchronous hardware driver and the NetNAT. See *Additional Information* for recommended interface naming conventions.

Command Syntax

```
attach asy [i/o] [irq] ppp [name] [window] [buf] [speed]
[opt]
```

Command Parameters

| Parameter Name | Description |
|----------------|--|
| i/o | The communication port I/O Address (e.g., 0x3f8 for com1, 0x2f8 for com2) |
| irq | The com port IRQ (e.g., 4 for com1, 3 for com2) |
| name | The name of the NetNAT's logical interface (typically pp0). |
| window | Use the value 8092 |
| buf | Use the value 1500 |
| opt | Optional. When used, select a value from the following list: <ul style="list-style-type: none">◆ 5 for 16550A◆ 6 for 16650◆ 7 for 16750 UART |

For Example

To attach the first (or only) ASYNC driver to NetNAT interface "pp0" at 230,400bps using a 16650, the command is:

```
attach asy 0x3f8 4 ppp pp0 8092 1500 230400 6
```

This attaches a PPP driver to interface "pp0" using a StarTech/Exar 16650 on COM1.

Possible responses are:

- ◆ "Interface xxx already exists" when the specified interface already identifies an attached hardware driver.
- ◆ No response when the command is successful.

auth

Purpose

The start auth global command starts/stops and controls the Authentication Server. This server responds to the 'who is on IP Address/Port nnn/n?' queries from telnet and other servers with canned information to satisfy the remote host, but without giving away any information about your users. The protocol is defined in RFC 1413. At startup, NetNAT sets the auth reply to "USERID:NSCUSER".

Command Syntax

```
start auth
stop auth
auth
auth hidden
auth user userid
```

Command Parameters

| Parameter Name | Description |
|----------------|--|
| hidden | Changes the auth reply to "ERROR:HIDDEN-USER." |
| user userid | Changes the auth reply to "USERID:userid," where userid is an arbitrary value that you provide. For example, "auth user NSCUSER" would set the reply string to the original default value. |

Additional Information

The BSD Authentication Server listens on port 113. It responds to queries from telnet and other servers. This service can reduce session startup time by up to 10 seconds by eliminating the remote system timeout awaiting an authentication response. The request consists of a tiny message containing the source port number of the user client application (or what the remote host thinks is the port number, when we're doing port translation). Our reply contains the specified port number and a canned user login ID of "NSCUSER," or a value that you specify. This satisfies the remote host while giving away no private information.

The stop auth command may be used to shut down the authentication server.

The "auth" command with no parameters will display the current value of the reply string.

In order for external hosts to access this authentication service, you must enable it as a "local" service with the "service local" command.

cd

Purpose

The `cd` command is equivalent to a unix or MS-DOS “change directory” command. It changes the directory for the NetNAT console commands, but does not change the “working directory” where the NetNAT configuration and web files are located.

Command Syntax

```
cd [path]
```

Command Parameters

There are none, other than the directory path.

delete

Purpose

The delete command is equivalent to the MS-DOS file deletion command.

Command Syntax

```
delete [filename]
```

Command Parameters

None, other than the name of the file to delete.

dhcp

Purpose

dhcp is the Distributed Host Configuration Protocol, and is used to assign IP Addresses and other information to hosts that are just coming online. This is a superset of an older, like facility called BOOTP (for the Bootstrap Protocol). DHCP is gaining popularity by easing the administrative load on MIS and network support personnel. The NetNAT has a DHCP client that allows you to solicit your public IP Address from an outside DHCP server. This is popular when the NetNAT is being used as a firewall in a cable TV cable modem Internet distribution. Subsequent sections will detail the sub-command functions.

Command Syntax

```
dhcp
dhcp [int name]
dhcp [int name] enable {hostname}
dhcp [int name] disable
dhcp [int name] release
dhcp [int name] renew
```

Command Parameters

| Parameter Name | Description |
|----------------|---|
| int name | The name of the NetNAT interface that is using DHCP. Any and all ethernet interfaces are capable and permitted to use DHCP. |
| hostname | The optional "system name" that is to be sent in DHCP packets. If not specified on the dhcp ... enable configuration line, the NetNAT will use the NetNAT's hostname instead. |

For Example

| | |
|----------|---|
| dhcp | Displays all available DHCP information for all interfaces that are using DHCP. |
| dhcp en0 | Displays all DHCP information for interface "en0." |

To display the DHCP information for interface en0 in DHCP "bound" mode, with a complete set of information from a server:

```
nscnat> dhcp en0
en0 BOUND, ClientIP=144.12.3.4 Server=144.12.5.1 Netmask=255.255.0.0
Nameservers=144.12.5.211,144.12.19.02 Gateway=144.12.19.254
Hostname=nscnat, Domain name=mumble.dom
Lease: Original=1:00:00:00 Remaining=0:06:14:10
T1=0:03:41:55 T2=0:00:52:04 LastXID=0xafd60522
nscnat> _
```

| Displayed Field | Description |
|-----------------|--|
| BOUND | This is the current state of the dhcp client for the specified interface. Possible values are IDLE, DISCOVERY, REQUEST, BOUND, RELEASING, RENEWING and REBINDING. Details on DHCP and the states are found below, and in RFC 1541. |
| ClientIP | The IP Address that this interface was given by the DHCP server or zero if none. |
| Server | The IP Address of the DHCP server that gave us our configuration or zero if none. |
| Netmask | The netmask that we received from the DHCP server. |
| Broadcast | The broadcast address that we received from the DHCP server. |
| Nameservers | The addresses of up to two DNS servers given to us by the DHCP server. |
| Gateway | Our default router or gateway as given to us by DHCP. |
| Hostname | Our host name. |
| Domain name | The domain name that we received from the DHCP server. |
| Lease | The length of time that the received information should be considered valid. After that time we may no longer use the information. |
| Original | The original length of our lease in days:hours:minutes:seconds. |
| Remaining | The amount of time remaining on our lease in the same units. |
| T1 | A timer that causes us to attempt to renew our lease when it expires. It is set initially to one-half of the original lease time. |
| T2 | A last-gasp timer that attempts to get our lease re-bound just before it expires. |
| LastXID | The 32-bit number included in the last DHCP message exchanges. This is for debugging only. |

The DHCP States

| DHCP State | Description |
|------------|--|
| IDLE | We are not trying to get new information from the DHCP server and have no configuration. |
| DISCOVERY | We are sending "Discovery" broadcast messages to solicit a configuration offer from one or more DHCP servers. |
| REQUEST | We have received one or more offers from DHCP servers, have selected an offer we like, and have sent a "Request" message for the information offered. |
| BOUND | We received an acknowledgement in response to our "Request," and are using the offered information. |
| RENEWING | Our T1 timer has expired, and we are attempting to renew our lease before it lapses. |
| REBINDING | We have been unable to renew our lease, and it's about to expire. We are sending "Discovery" packets in a frantic attempt to get a new lease before our information expires. |
| RELEASING | We no longer want the information we received and have sent a "Release" message to surrender the information. |

dhcp ... enable

Purpose

dhcp enable establishes a DHCP process for the specified interface. All existing address information is deleted from the interface in anticipation of receiving fresh information from a DHCP server. An optional “system name” may be specified in this sub-command to override the DHCP client’s use of your NetNAT’s hostname in DHCP messages.

Command Syntax

```
dhcp [int name] enable {hostname}
```

Command Parameters

| Parameter Name | Description |
|----------------|---|
| int name | The name of the NetNAT interface that is using DHCP. Any and all ethernet interfaces are capable and permitted to use DHCP. |
| hostname | The optional “system name” that is to be sent in DHCP packets. If not specified on the dhcp ... enable configuration line, the NetNAT will use the NetNAT’s hostname instead. |

For Example

| | |
|-----------------|---|
| dhcp en0 enable | Enables DHCP operation on ethernet interface “en0.” |
|-----------------|---|

To start DHCP on interface en0 (overriding the system name) and then display the result before any server can reply:

```
nscnat> dhcp en0 enable cx29292-12
nscnat> dhcp en0
en0 DISCOVERY, ClientIP=0.0.0.0 Server=0.0.0.0 Netmask=0.0.0.0
Nameservers=0.0.0.0,0.0.0.0 Gateway=0.0.0.0
Hostname=cx29292-12, Domain name=(null)
Lease: Original=0:00:00:00 Remaining 0:00:00:00
T1=0:00:00:00 T2=0:00:00:00 LastXID=0xafd60522
nscnat> _
```

dhcp ... disable

Purpose

dhcp disable terminates the DHCP process for the specified interface.

Command Syntax

dhcp [int name] disable

Command Parameters

| Parameter Name | Description |
|----------------|--|
| int name | The name of the NetNAT interface that is using DHCP. Any and all ethernet interfaces are capable and permitted to use DHCP. |

For Example

| | |
|------------------|--|
| dhcp en0 disable | Disables DHCP operation on ethernet interface "en0." |
|------------------|--|

dhcp ... release

Purpose

dhcp release clears previously-acquired IP information, giving up the data received from a DHCP server. This may be done in preparation for re-acquiring DHCP information or for testing.

Command Syntax

```
dhcp [int name] release
```

Command Parameters

| Parameter Name | Description |
|----------------|---|
| int name | The name of the NetNAT interface that is using DHCP. Any and all ethernet interfaces are capable and permitted to use DHCP. |

For Example

| | |
|------------------|---|
| dhcp en0 release | Forget all acquired DHCP information on ethernet interface "en0." |
|------------------|---|

dhcp ... renew

Purpose

dhcp renew contacts the DHCP server that configured the NetNAT's interface and requests a renewal of the acquired information. This normally happens half way through the lease period, or may be initiated manually with this sub-command.

Command Syntax

```
dhcp [int name] renew
```

Command Parameters

| Parameter Name | Description |
|----------------|--|
| int name | The name of the NetNAT interface that is using DHCP. Any and all ethernet interfaces are capable and permitted to use DHCP. |

For Example

| | |
|----------------|--|
| dhcp en0 renew | Attempts to refresh previously-received DHCP information for ethernet interface "en0." |
|----------------|--|

dir

Purpose

The `dir` command is similar to an MS-DOS “directory” command. It always displays all available file information. There are no optional arguments other than specification of the file to display.

Command Syntax

```
dir {pattern}
```

Command Parameters

None, other than the optional file name or wildcard pattern.

discard

Purpose

The discard global command starts the discard server. This server accepts and throws away any packets received.

Command Syntax

```
start discard
```

Command Parameters

None

For Example

```
start discard
```

domain

Purpose

The domain commands control the NetNAT's use of the Domain Name Service. There is usually no need for your NetNAT to know anything about DNS, since its role is that of firewall rather than host. Because of this, many of these commands will be undocumented due to laziness.

Command Syntax

```
domain addserver [ip addr]
domain dropserver [ip addr]
domain list
domain query [hostname]
domain retry [interval]
domain suffix [domain name]
domain cache clean
domain cache list
domain cache size [size]
domain cache wait [interval]
```

Command Parameters

| Parameter Name | Description |
|----------------|---|
| ip addr | An IP Address of a domain server. |
| hostname | A fully-qualified domain name for a host. |
| interval | A time period in seconds. |
| size | Number of cache entries to keep in memory (default 20). |

For Example

```
domain suffix safety.net
```

domain addserver

Purpose

The domain addserver sub-command gives your NAT knowledge of a domain name server for its own use. There is usually no need for your NetNAT to know anything about DNS, since its role is that of firewall rather than host.

Command Syntax

```
domain addserver [ip addr]
```

Command Parameters

| Parameter Name | Description |
|----------------|-----------------------------------|
| ip addr | An IP Address of a domain server. |

For Example

```
domain addserver 192.128.133.151
```

This gives your NetNAT the address of a computer that will respond to DNS queries. The NetNAT does not need this for normal operation.

domain dropserver

Purpose

The domain dropserver sub-command deletes knowledge of a particular domain name server. There is usually no need for your NetNAT to know anything about DNS, since its role is that of firewall rather than host.

Command Syntax

```
domain dropserver [ip addr]
```

Command Parameters

| Parameter Name | Description |
|----------------|--|
| ip addr | An IP Address of a domain server previously defined with a “domain addserver” command. |

For Example

```
domain dropserver 192.128.133.151
```

This deletes the specified name server from the NetNAT’s tables.

domain list

Purpose

The domain list sub-command displays all information known by the NetNAT relative to domain name servers. There is usually no need for your NetNAT to know anything about DNS, since its role is that of firewall rather than host.

Command Syntax

domain list

Command Parameters

None.

For Example

```
nscnat> domain list
Server address      srtt  mdev  timeout  queries  responses
192.128.133.151    197   27    2003     21       21
nscnat> _
```

| Displayed Field | Description |
|-----------------|---|
| Server address | The IP Address of a configured Domain Name Server. |
| srtt | The “smoothed” measured response time between the NetNAT and the specified Domain Name Server. The initial value is 2000 milliseconds, which will change as more queries are processed. |
| mdev | Mean deviation in response time from the specified Domain Name Server. |
| timeout | The timeout value in milliseconds for the specified DNS host. |
| queries | The number of DNS queries from the NetNAT to the specified DNS host. |
| responses | The number of DNS responses received by the NetNAT from the specified DNS host. |

echo

Purpose

The echo global command starts the echo server. This server responds to ping requests.

Command Syntax

```
start echo
```

Command Parameters

None

For Example

```
start echo
```

Additional Information

This server handles “ping” requests that are addressed to the NetNAT itself. If a ping is addressed to one of the many possible sets of addresses that the Internet may access through the NetNAT, a complex set of rules are applied to determine whether the NetNAT should answer, ignore the message, or pass it through for an Intranet computer to handle.

exit

Purpose

The `exit` global command causes the NetNAT to kill all sub-processes, release all conventional and XMS memory, and exit to DOS. Before quitting, it disables the watchdog timer.

Command Syntax

exit

Command Parameters

None

For Example

exit

gaming

Purpose

The gaming command displays and may modify the state of the Multi-point UDP Gaming switch. Enabling multi-point gaming, will permit an internal client to converse with more than one destination hosts through a single UDP session context object. Many multi-player Internet games require this facility to work. This is enabled by default.

Command Syntax

gaming {on|off}

Command Parameters

| Parameter Name | Description |
|----------------------|--|
| on true yes 1 | Enable the Multi-point UDP Gaming facility. |
| off false no 0 | Disable the Multi-point UDP Gaming facility. |

There is no response when the command parameters are correct.

With no parameter, the gaming command will display the state of the Multi-point UDP Gaming switch.

hop

Purpose

The hop command is a debugging tool for the network administrator. It is equivalent to the unix “traceroute” and Windows “tracert” commands.

Command Syntax

```
hop check [hostid]
hop maxttl [max hops]
hop maxwait [max time]
hop queries [# queries]
```

Command Parameters

| Parameter Name | Description |
|----------------|--|
| hostid | The hostname or IP Address of the specified destination. |
| max hops | The maximum value to place in the Time To Live (TTL) field of the IP headers of hop check queries. If not specified, the NetNAT will default to 30 hops. |
| max time | Maximum time that the NetNAT should wait for replies. The default is 5 seconds. |
| # queries | The number of queries to be sent at each hop level. The default is 3 queries per hop level. |

hop check

Purpose

The hop check command is a debugging tool for the network administrator. It is equivalent to the unix “traceroute” and Windows “tracert” commands. Like the unix traceroute, it uses UDP packets addressed to high-numbered ports and small Time-to-Live values to discover the path that messages follow to a remote destination. Each hop along the way should return an ICMP TTL Exceeded error message. The destination computer should return an ICMP Port Unreachable message.

Command Syntax

```
hop check [host id]
```

Command Parameters

| Parameter Name | Description |
|----------------|--|
| hostid | The hostname or IP Address of the specified destination. |

For Example

Trace the route to the remote computer fred.bogus.dom:

```
nscnat> hop check fred.bogus.dom
Resolving fred.bogus.dom... 204.167.96.12:33434
 1: 10.1.16.254  mygw.my.dom. (11 ms) (9 ms) (9 ms)
 2: 10.14.27.254 corpgw.my.dom. (16 ms) (15 ms) (15 ms)
 3: 207.86.23.3  ispgw.isp.dom. (31 ms) (34 ms) (41 ms)
 4: 207.87.33.19 hssil.den.isp.dom. (86 ms) (91 ms) (87 ms)
 5: 207.87.33.24 isp.den.oisp.dom. (91 ms) *** (93 ms)
 6: 197.41.59.77 fddil.sjc.oisp.dom. (110 ms) (119 ms) (106 ms)
 7: 197.41.59.21 bogusgw.oisp.dom. (133 ms) (142 ms) (138 ms)
 8: 204.167.96.12 fred.bogus.dom. (140 ms) (140 ms) (148 ms)
traceroute done: normal (Unreachable Port)
Hit enter to continue
nscnat> _
```

hostname

Purpose

The hostname global command sets a unique name for the NetNAT. This name will be the command prompt on the console, and is the default system name when you use DHCP to solicit IP Addresses and gateway information.

Command Syntax

hostname [hostname]

Command Parameters

| Parameter Name | Description |
|----------------|--|
| hostname | A string value that uniquely identifies your NetNAT. |

For Example

To name your NetNAT device 'MyNAT', the command is:

```
fw> hostname MyNAT
MyNAT> _
```

icmp

Purpose

The ip commands display or modify some aspect of the NetNAT's handling of IP packets. The interesting one is ip status which displays the contents of the IP "MIB" (the SNMP Management Information Block containing the IP statistics).

Command Syntax

icmp status

Command Parameters

None.

Possible responses are:

- ◆ Display of IP statistics.

```
nscnat> icmp status
( 1)icmpInMsgs          28      (14)icmpOutMsgs          0
( 2)icmpInErrors        0      (15)icmpOutErrors        0
( 3)icmpInDestUnreachs  4      (16)icmpOutDestUnreachs  0
( 4)icmpInTimeExcds    24      (17)icmpOutTimeExcds    0
( 5)icmpInParmProbs     0      (18)icmpOutParmProbs     0
( 6)icmpInSrcQuenchs    0      (19)icmpOutSrcQuenchs    0
( 7)icmpInRedirects     0      (20)icmpOutRedirects     0
( 8)icmpInEchos         0      (21)icmpOutEchos         0
( 9)icmpInEchoReps      0      (22)icmpOutEchoReps      0
(10)icmpInTimestamps   0      (23)icmpOutTimestamps   0
(11)icmpInTimestampReps 0      (24)icmpOutTimestampReps 0
(12)icmpInAddrMasks    0      (25)icmpOutAddrMasks    0
(13)icmpInAddrMaskReps 0      (26)icmpOutAddrMaskReps 0
nscnat> _
```

ipInReceives shows the number of inbound IP packets that the NetNAT has received, ipInHdrErrors contains the number of header format or checksum errors seen, ipForwDatagrams counts packets forwarded through the NetNAT (whether translated or not), ipInDelivers and ipOutRequests shows packets either were accepted for local processing or that originated in the NetNAT.

ifconfig

Purpose

The ifconfig interface command displays summary status information and counts for a specific interface or all interfaces attached to this NetNAT. The ifconfig sub-commands (see *Additional Information* below) allow you to set operational parameters for each logical interface you create with the attach command.

Command Syntax

```
ifconfig [int name]
```

Command Parameters

| Parameter Name | Description |
|----------------|---|
| blank | Leave the interface name blank to display status information for all 'attached' interfaces. |
| int name | To view status information for a specific 'attached' interface, indicate the appropriate interface name (e.g. en0). |

For Example

To view status information for all interfaces attached to this NetNAT, the command is:

```
ifconfig
```

To view status information for interface en0, the command is:

```
ifconfig en0
```

Shown below is an example from the interface status display for a public interface.

```
nscnat> ifconfig en0
en0 IP addr 204.17.3.5 MTU 1550 Link encap Ethernet
Link addr 00:20:af:d7:05:27, MC forwarding OK, 0 groups
trace 0x0 trclim 0 netmask 0xffffffff broadcast 204.17.3.255
sent: ip=3314 tot=3316 pps=0.10 mpps=16.50 idle=0:00:00:09
recv: ip=3028 tot=5337 pps=0.20 mpps=11.70 idle=0:00:00:01
```

Meanings for the various fields will be shown after the next example.

ifconfig en1

Shown below is an example from the interface status display for a private interface. Besides the obviously private IP Address (from the 10-net), the word “Private” at the end of the first line is the giveaway.

```
nscnat> ifconfig en1
en1 IP addr 10.1.1.254 MTU 1550 Link encap Ethernet Private
Link addr 00:60:af:d7:35:2b, MC forwarding OK, 0 groups
trace 0x0 trclim 200 netmask 255.255.255.0 broadcast 10.1.1.255
sent: ip=3811 tot=3964 pps=0.20 mpps=11.50 idle=0:00:00:01
recv: ip=3751 tot=6379 pps=0.40 mpps=16.60 idle=0:00:00:09
```

Displayed Field Definitions

The displayed fields are as follows:

| Field Name | Description |
|----------------------|---|
| IP addr | The main IP Address for the interface. If it's a public interface, it may have lots of other IP Addresses that it will respond to. See the service mapping commands for more information. |
| MTU | The largest block this interface will transmit. |
| Link encap | The encapsulation in use on this interface, usually “Ethernet.” |
| Private | Present to indicate a private or internal network. Missing represents a public interface. |
| Link addr | The hardware or “MAC” address of the interface card. |
| MC | The multicast capabilities of the interface. The possible capabilities: <ul style="list-style-type: none">• forwarding OK – interface capable of multicast forwarding.• not forwarded – multicast capable but not to be forwarded.• not possible – the interface card isn't capable of multicast.• not permitted – multicast is not permitted. |
| groups | When multicast is permitted, this shows the number of multicast groups that the NetNAT has detected via IGMP. |
| trace | The trace flags. Zero means no tracing active. Non-zero shows the flags used when the trace was started. See the ‘trace’ command. |
| trclim | The number of packets at which the trace will stop automatically, or zero if this feature is disabled. |
| netmask | The netmask for this interface in dotted-decimal representation. |
| broadcast | The broadcast address for this interface. |
| Interface visible... | Present on private interfaces to remind you that a NetNAT with more than one private interface simply routes between those, providing no firewall protection. |
| sent: | Begins the transmit packet statistics line. |
| recv: | Begins the receive packet statistics line. |
| ip= | The total number of IP packets transmitted or received. |
| tot= | The total number of packets of any type sent or received. |
| pps= | The average packets per second for the most recent 10-second period. This is updated every ten seconds. |
| mpps= | The maximum pps so far: the largest value of pps seen since the NetNAT was started. |
| idle= | The time in days:hours:minutes:seconds since the last packet was |

sent or received.

Additional Information

The ifconfig sub-commands set the parameters for each logical interface you create with the attach command. The ifconfig sub-commands are:

| | |
|------------|--|
| ipaddress | Sets the apparent IP address for the Interface. |
| netmask | Sets the netmask for the network to which the NetNAT is connected. |
| broadcast | Sets the Broadcast Address for the network. |
| private | Sets the Private or Public designation for the network. |
| Tracelimit | Sets the maximum number of packet trace items before packet tracing is automatically stopped on the interface (to keep from filling all available disk space). |

You must issue appropriate ifconfig sub-commands for each interface you create with the attach command. Described below are the command syntax and parameters for each ifconfig sub-command.

ifconfig -- ipaddress

Purpose

The `ifconfig ipaddress` sub-command sets the apparent IP address for the specified network interface (e.g., `en0`).

Command Syntax

```
ifconfig [int name] ipaddress [ip address]
```

Command Parameters

| Parameter Name | Description |
|----------------|--|
| Int name | The name of the NetNAT logical interface (e.g. <code>en0</code>). |
| IP Address | The apparent IP Address for this interface in dotted-decimal notation. |

Possible responses are:

- ◆ "Interface xxx unknown" when there has not been an `attach` command issued for the specified interface.
- ◆ No response when the command is successful.

For Example

To set the IP address `123.234.222.17` for the logical interface `en0`, the command is:

```
ifconfig en0 ipaddress 123.234.222.17
```

Additional Information

The NetNAT uses the IP Address when:

- ◆ Originating messages
- ◆ Acting as a gateway to and from the internet for hosts on the private intranet.
- ◆ Performing dynamic service Mapping duties

ifconfig ... broadcast

Purpose

The ifconfig broadcast sub-command sets the broadcast IP address for the specified interface (e.g., en0).

Command Syntax

```
ifconfig [int name] broadcast [ip address]
```

Command Parameters

| Parameter Name | Description |
|----------------|---|
| Int name | The name of the NetNAT logical interface (e.g. en0). |
| IP Address | The broadcast IP Address for this interface in dotted-decimal notation. |

Possible responses are:

- ◆ "Interface xxx unknown" when there has not been an attach command issued for the specified interface.
- ◆ No response when the command is successful.

For Example

To set the broadcast IP address 123.234.222.255 for the logical interface en0, the command is:

```
ifconfig en0 broadcast 123.234.222.255
```

Additional Information

When an interface is created by the attach command and given an IP Address by the ifconfig ipaddress sub-command, the broadcast address is set to the "all-ones" broadcast address for the combination of IP Address and netmask. In the example above, if the IP Address and netmask for interface en0 were 123.234.222.x and 255.255.255.0, the ifconfig broadcast sub-command would be unnecessary, since the NetNAT would have configured 123.234.222.255 as the broadcast address already. The explicit configuration of the broadcast address is not a "bad idea," since it makes very clear what you want the broadcast address to be.

ifconfig ... netmask

Purpose

The `ifconfig netmask` sub-command identifies the netmask for the network attached to the specified NetNAT interface (e.g., `en0`).

Command Syntax

```
ifconfig [int name] netmask [netmask]
```

Command Parameters

| Parameter Name | Description |
|----------------|--|
| int name | The name of the NetNAT logical interface (e.g. <code>en0</code>). |
| Netmask | The 32-bit netmask for this interface in dotted-decimal notation (e.g., <code>255.255.255.0</code>) |

Possible responses are:

- ◆ "Interface xxx unknown" when there has not been an `attach` command issued for the specified interface.
- ◆ No response when the command is successful.

For Example

To set the netmask for a non-segmented Class C network, for the logical interface `en0`, the command is:

```
ifconfig en0 netmask 255.255.255.0
```

Additional Information

All workstations, hosts, and routers use the 32-bit netmask to determine when two IP addresses are in the same logical network. All devices on a given network should use the same netmask value (e.g. `255.255.255.0`).

Note: The NetNAT does not support 'comb' style netmasks. All one-bits in the netmask must be contiguous.

ifconfig ... private

Purpose

The `ifconfig private` sub-command indicates how the interface will handle network address translation.

Command Syntax

```
ifconfig [int name] private [0|1|2]
```

Command Parameters 1

| Parameter Name | Description |
|----------------|---|
| int name | The name of the NetNAT logical interface (e.g. en0). |
| 0 1 2 | The optional values for the private parameter. 0 - This is a public network that uses IP addresses known to other public or Enterprise networks. 1 - This is a private network that will communicate directly with another private network and will not use address translation. 2 - This is a private network that should be concealed from other private networks as well as from public networks. |

Possible responses are:

- ◆ "Interface xxx unknown" when there has not been an `attach` command issued for the specified interface.
- ◆ No response when the command is successful.

For Example

To mark this interface as the public interface (the one that faces toward the Internet), the command is:

```
ifconfig en0 private 0
```

To mark this interface as one of your private interfaces (one that faces toward your internal Intranet), the command is:

```
ifconfig en0 private 1
```

To communicate with public networks using address translation and apparent IP addresses, concealing the actual IP address of the hosts on your private network, the command is:

ifconfig ... tracelimit

Purpose

The `ifconfig tracelimit` sub-command sets a maximum packet trace file size in an indirect way. The specified limit is number of traced packets. When running packet traces (captures) on this interface, the trace mechanism will stop tracing automatically when the specified number of packets have been traced. The trace file (if tracing to disk) will be closed for retrieval and examination.

Command Syntax

```
ifconfig [int name] tracelimit [0 | number]
```

Command Parameters

| Parameter Name | Description |
|----------------|--|
| int name | The name of the NetNAT logical interface (e.g. en0). |
| 0 number | The number of packets or zero to remove the limit. |

Possible responses are:

- ◆ “Interface xxx unknown” when there has not been an `attach` command issued for the specified interface.
- ◆ “Removed interface trace limit” when zero is specified as a limit.
- ◆ “Setting interface trace limit to nnn lines” where “nnn” is the specified number of packets.

For Example

To limit the number of captured packets to 200 on interface `en0`, the command is:

```
ifconfig en0 tracelimit 200
```

To remove a previously-imposed `tracelimit`, the command is:

```
ifconfig en0 tracelimit 0
```

ifconfig ... vlan

Purpose

The `ifconfig vlan` sub-command identifies an ethernet interface as a VLAN Trunk. As such, it may have sub-interfaces that appear to be separate ethernet interfaces in separate subnets. See the `attach vlan` command for the mechanism used to create sub-interfaces.

Command Syntax

```
ifconfig [int name] vlan [0 | 1]
```

Command Parameters

| Parameter Name | Description |
|----------------|--|
| int name | The name of the NetNAT logical interface (e.g. en0). |
| 0 1 | A flag to indicate true or false. 1 means true. |

Possible responses are:

- ◆ "Interface xxx unknown" when there has not been an `attach` command issued for the specified interface.

igmp

Purpose

The `igmp status` command displays the contents of the IGMP “MIB” (the SNMP Management Information Block containing the IGMP statistics). IGMP is the Internet Group Management Protocol, which controls the forwarding of multicast IP traffic through routers. It is used to detect and maintain lists of multicast groups on each interface of a router (or in this case, of the NetNAT). If IP Multicast Forwarding is enabled, multicast messages from a group on one interface are forwarded out other interfaces that also have members of that group. Multicast forwarding is by default disabled on the NetNAT. Enabling it will permit multicast traffic to pass through your NetNAT Firewall, and may constitute a breach of network security. Please give this careful thought.

Command Syntax

| | |
|------------------------------------|---|
| <code>igmp status</code> | Display the IGMP statistics as maintained in the IGMP MIB. |
| <code>igmp forward {on off}</code> | Display and optionally modify the state of the Multicast Forwarding switch. |
| <code>igmp trace {on off}</code> | Display and optionally modify the state of the IGMP trace flag. |

Command Parameters

Depends on the subcommand.

Possible responses to the IGMP status command are:

- ◆ Display of IGMP statistics.

```
nscnat> igmp status
( 1)igmpInMsgs      0      ( 8)igmpOutMsgs      0
( 2)igmpInErrors    0      ( 9)igmpOutErrors    0
( 3)igmpInBcsts     0      (10)igmpOutBcsts     0
( 4)igmpInQueries   0      (11)igmpOutQueries   0
( 5)igmpInV1Reports 0      (12)igmpOutV1Reports 0
( 6)igmpInV2Reports 0      (13)igmpOutV2Reports 0
( 7)igmpInLeaves    0      (14)igmpOutLeaves    0
nscnat> _
```

igmp forward

Purpose

The `igmp forward` command displays and may modify the state of the IP Multicast Forwarding switch. Enabling Multicast Forwarding will cause multicast messages to be forwarded through your NetNAT under control of the Internet Group Management Protocol (IGMP). This is disabled by default. We recommend that you give careful thought to the impact on network security of enabling this function.

Command Syntax

```
igmp forward {on|off}
```

Command Parameters

| Parameter Name | Description |
|----------------------|---|
| on true yes 1 | Enable forwarding of Multicast traffic between IGMP Multicast Groups on NetNAT interfaces. |
| off false no 0 | Disable forwarding of Multicast traffic between IGMP Multicast Groups on NetNAT interfaces. |

There is no response when the command parameters are correct.

igmp trace

Purpose

The `igmp trace` command displays and may modify the state of the IGMP Event Tracing switch. Enabling IGMP tracing will cause Internet Group Management Protocol (IGMP) messages (like group queries and membership reports) to be decoded and displayed on the NetNAT console for diagnostic or educational purposes. This is disabled by default.

Command Syntax

```
igmp trace {on|off}
```

Command Parameters

| Parameter Name | Description |
|----------------------|--|
| on true yes 1 | Enable decoding and display of IGMP messages on the NetNAT console. |
| off false no 0 | Disable decoding and display of IGMP messages on the NetNAT console. |

There is no response when the command parameters are correct.

ip

Purpose

The ip commands display or modify some aspect of the NetNAT's handling of IP packets. The interesting one is ip status which displays the contents of the IP "MIB" (the SNMP Management Information Block containing the IP statistics).

Command Syntax

ip status

Command Parameters

None.

Possible responses are:

- ◆ Display of IP statistics.

```
nscnat> ip status
( 1)ipForwarding          1      ( 2)ipDefaultTTL        255
( 3)ipInReceives         3523   ( 4)ipInHdrErrors       8
( 5)ipInAddrErrors       0      ( 6)ipForwDatagrams     3272
( 7)ipInUnknownProtos    0      ( 8)ipInDiscards       0
( 9)ipInDelivers         242    (10)ipOutRequests       322
(11)ipOutDiscards        0      (12)ipOutNoRoutes      0
(13)ipReasmTimeouts      30     (14)ipReasmReqds       0
(15)ipReasmOKs          0      (16)ipReasmFails       0
(17)ipFragOKs           0      (18)ipFragFails       0
(19)ipFragCreates        0
Routing lookups: 5669, cache hits 5662 (100%)
nscnat>
```

ipInReceives shows the number of inbound IP packets that the NetNAT has received, ipInHdrErrors contains the number of header format or checksum errors seen, ipForwDatagrams counts packets forwarded through the NetNAT (whether translated or not), ipInDelivers and ipOutRequests shows packets either were accepted for local processing or that originated in the NetNAT.

memory

Purpose

The memory commands display or modify the characteristics and status of the memory heap and network controller interrupt circular buffer. The sub-commands are:

Command Syntax

| | |
|-----------------|--|
| memory freelist | Display the free areas of heap memory. |
| memory sizes | Displays the sizes of requested memory buffers, including special message buffers. |
| memory status | Display the heap size, available heap memory, message buffer allocation info and other data. |

memory freelist

Purpose

The memory freelist sub-command displays the status of the memory heap free memory list. This shows the blocks of memory that are available for allocation to NetNAT processes. The NetNAT allocator tries to combine freed memory blocks into larger blocks to reduce fragmentation of the heap.

Command Syntax

| | |
|-----------------|--|
| memory freelist | Display the free areas of heap memory. |
|-----------------|--|

For Example

```
nscnat> memory freelist
7b910000 16 | 7c2f0000 32 | 7c380008 96 | 7c4b0008
104
7c5f0000 64 | 7dae0008 152 | 7dc60000 32 | 7e280008
624
7e710008 1032 | 7efc0008 584 | 7f6a0008 1576 | 80cd0008
10664
85e30000 4008 |
nscnat>
```

memory sizes

Purpose

The memory sizes sub-command displays a bucket-style (histogram) summary of the sizes of memory buffers that have been requested by the NetNAT processes. This is purely informational, and allows the programmer to find frivolous heap users

Command Syntax

| | |
|--------------|---|
| memory sizes | Display a summary of the sizes of memory buffers requested by the NetNAT processes. |
|--------------|---|

For Example

```
nscnat> memory sizes
N>= 1: 0 | N>= 2: 0 | N>= 4: 31 | N>= 8: 156
N>= 16: 633 | N>= 32: 363 | N>= 64: 236 | N>= 128: 34
N>= 256: 229 | N>= 512: 279 | N>= 1024: 113 | N>= 2048: 43
N>= 4096: 8 | N>= 8192: 0 | N>=16384: 0 | N>=32768: 0
nscnat>
```

memory status

Purpose

The memory status sub-command a bucket-style (histogram) summary of the sizes of memory buffers that have been requested by the NetNAT processes. This is purely informational, and allows the programmer to find frivolous heap users

Command Syntax

| | |
|---------------|--|
| memory status | Display a summary of the memory allocator statistics |
|---------------|--|

For Example

```
nscnat> memory status
heap size 91920 avail 28832 (31%) morecores 110 coreleft
88656
allocs 2000 frees 1823 (diff 177) alloc fails 0 invalid frees
0
garbage collections yellow 0 red 0

mbuf allocs 13838 free cache hits 13817 (99%) mbuf frees
13837
pushdown calls 23967 pushdown calls to alloc_mbuf 3210
Free cache: small 8 medium 3 large 9
nscnat> _
```

The first line shows the total size, the amount of heap memory available in bytes and as a percentage of the total heap size, and the amount of memory left over (i.e. not placed on the heap at startup) and therefore available for shell subcommands.

The second line shows the total number of calls to allocate and free blocks of memory, the difference of these two values (i.e. the number of allocated blocks outstanding), the number of allocation requests that were denied due to lack of memory, and the number of calls to 'free' that attempted to free garbage (e.g. by freeing the same block twice or freeing a garbled pointer).

The third line shows the “garbage collection” status. Garbage collection is a last-ditch process to reclaim as much memory as possible when we’re in trouble.

An “mbuf” is a specialized memory buffer for network messages. There are just three sizes of mbufs, and an mbuf request is rounded up to the next larger size. The reason for this is twofold: to keep from fragmenting memory with allocations of various sizes of buffers, and to make allocation and release of message buffers very efficient. Once an mbuf has been created, it is not freed back to the heap, but is rather chained onto one of three mbuf queues (based on size). When the network process needs an mbuf, the alloc_mbuf routine looks on the queues (called the mbuf caches) for one. If none are found there, the normal heap allocator is called to create a new one.

The fourth line contains information about the mbuf allocation process. The total number of allocations (allocs) is shown, as well as the number of times that we looked for an mbuf and found one on the “free cache” (the queue that we keep mbufs on when they’re not in use). The number of mbuf frees is shown as well.

“Pushdowns” are where we need to put information (usually headers) on the front of an existing message. If the mbuf containing the message has sufficient room in front of the message data, then the information may be placed in the existing mbuf. We try very hard to make this work this way, since having to allocate a buffer to hold the new information is very inefficient.

The fifth line contains pushdown information. The first part shows the number of pushdowns we have used. Pushdowns are an integral part of our system, so we should see lots. The next part contains the number of times that a pushdown call resulted in a call to `alloc_mbuf` to create a new buffer. This is very inefficient, so we strive to prevent it. It’s not awful, so don’t be alarmed if the number is large. It should be a tiny fraction of the total number of pushdowns.

The last line give us the total available mbufs in the three sizes that we use. Small mbufs will hold up to 32 bytes of data. Medium mbufs will hold up to 128 bytes, while the large ones will hold up to 2048 bytes. The number of free mbufs in each size is shown on the last line. The total number in existence will not be shown. It’s important to remember that we can create new mbufs from the heap. It’s also important to remember that we can add to the heap from the PC’s memory.

mkdir

Purpose

The mkdir command creates an MS-DOS directory with the specified name.

Command Syntax

| | |
|--------------|-------------------------------|
| mkdir [name] | Make the specified directory. |
|--------------|-------------------------------|

more

Purpose

More is a file pager, not unlike the unix “more” or an enhanced version of the MS-DOS “more” facility. More creates a new console window that shows the contents of the requested file, and allows you to scroll up and down through it. To exit the more window, type q or control-C.

Command Syntax

| | |
|-----------------|---------------------------------------|
| more [filename] | Display a file in a scrolling window. |
|-----------------|---------------------------------------|

Additional Information

The normal PC PgUp, PgDn, Home, End, and up-down arrow keys let you scroll around in the specified file. Dedicated emacs and vi addicts will be pleased to discover that their favorite up-down line keys (and a few others) also work with the more command.

page

Purpose

“Page” provides you a unix-like facility to execute a command and pipe its output to a pager. Page creates a new console window that shows the output of the executed command, and allows you to scroll up and down through it. To exit the page window, type q or control-C.

Command Syntax

| | |
|----------------|--|
| page [command] | Execute the specified NetNAT command and pipe the output to a pager in a new console window. |
|----------------|--|

Additional Information

For a discussion of the key-bindings for page up/down, etc., please refer to the more command section.

ping

Purpose

The ping command sends an ICMP echo request packet to the specified destination host. The round-trip-time is displayed when the response packet returns. If no response returns, there is no displayed result. Ping spawns a subtask to perform DNS lookup and response capture, so may not be run from a web browser command.

Command Syntax

```
ping [host] {length {interval {incflag}}}
```

Command Parameters

| Parameter Name | Description |
|----------------|--|
| host | Name or IP Address of target host to ping. |
| length | Length in bytes of the data section of the ping (default is 64). |
| interval | Interval between pings when you wish the NetNAT to ping until interrupted, expressed in milliseconds (1000 equals one second). |
| incflag | If present will cause the NetNAT to increment the host IP Address after each ping, resulting in the pinging of a range of hosts. |

For Example

```
nscnat> ping 10.1.1.6
10.1.1.6: rtt 9
nscnat> _
```

pptp

Purpose

The pptp command displays any PPTP connections that are passing through the NetNAT at this time.

Command Syntax

pptp

Command Parameters

None.

For Example

```
nscnat> pptp
Interface en0: 5 Connections
10.1.1.22  posing as 133.19.76.3  talking to 204.17.96.3
  p_in 101, p_out 100, b_in 12044, b_out 10771
10.1.1.19  posing as 133.19.76.3  talking to 167.16.116.9
  p_in 2101, p_out 2375, b_in 37115, b_out 39621
10.3.55.36 posing as 133.19.76.3  talking to 18.121.3.72
  p_in 4496, p_out 6432, b_in 20805, b_out 69927
10.4.101.1 posing as 133.19.76.4  talking to 138.121.37.15
  p_in 15128, p_out 15011, b_in 7840, b_out 98866
10.7.13.7  posing as 133.19.76.5  talking to
192.155.133.44
  p_in 1812, p_out 1900, b_in 9125, b_out 122873
nscnat> _
```

Displayed Information

There will be no displayed information if there are no active PPTP connections. When active PPTP connections exist, the actual IP Address of a computer, its apparent IP Address and the address of the remote computer will be displayed as above, plus counts of packets in and out, and byte counts in and out.

Additional Information

There may not be two connections with identical combinations of apparent and remote addresses. If you need more than one connection to a single remote computer, you should manipulate your NetNAT configuration to force a different apparent IP Address for the internal computer desiring the PPTP connection (see the service subnet command).

private

Purpose

The private command displays the current dynamically-created sessions between computers in your private Intranet and computers on the outside. Displayed information includes the actual internal IP Address and port number, the apparent IP Address and port number, the remote IP Address and port number, the protocol in use (TCP, UDP or ICMP), and the current status of the session.

Command Syntax

private

Command Parameters

None.

For Example

To view all the current session information, the command is:

```
nscnat> private
prot      Real IP.port Posing as IP.port  Dest IP.port      Status
ICMP      10.1.1.6:0   204.162.22.9:0    167.16.122.8:0   Dying
UDP       10.1.1.6:1900 204.162.22.9:2076 113.6.81.38:53   Dying
TCP       10.2.22.3:2511 204.162.22.9:2052 207.16.33.9:23
Active
TCP       10.2.22.3:2512 204.162.22.9:2099 207.16.33.9:23   Old
nscnat> _
```

Displayed Information

| | |
|-------------------|---|
| Prot | The protocol: ICMP, TCP or UDP. |
| Real IP.port | The actual IP Address and port number being used by the internal client computer. |
| Posing as IP.port | The apparent IP Address and port number as viewed from the outside. |
| Dest IP.port | The IP Address and port number of the server at the remote location. |
| Status | The current status of this session. |

Additional Information

The status displayed may mean different things for different protocols. The rules that make a session's status change from one state to another are many. In addition, the private command has a sub-command to set the idle timeout for an "old" session. This idle timeout is the time that a session of a particular type (UDP

or TCP) is allowed to exist with no activity. When the timer expires, the session is released. The states and their meanings are:

| | |
|--------|---|
| Active | The session has seen message activity within the last 10 seconds. |
| Dying | The session meets the qualifications to be released. If it sees no more activity in the next 10 seconds, it will disappear. |
| Old | The session has seen no activity within the past 10 seconds. Its idle timer is now running so that it may eventually be released after a reasonable amount of time. |

ICMP sessions are generally single-packet events, so are marked as “dying” immediately. TCP sessions that have passed packets with either the RST or FIN flags set are marked as “dying” immediately. In either case, continued activity will “push out” the 10-second timeout that releases them. SNMP trap and GET/Response sessions that the NetNAT translates are marked as dying. Doing this for this group of packets keeps our session usage low with a corresponding reduction in overhead.

The private `idle` sub-command to set the release timeout for TCP and UDP sessions is described in the next section.

private idle

Purpose

The `private idle` sub-command sets the release timeouts for UDP and TCP sessions. When no activity has occurred on a UDP or TCP session for this amount of time, the session context object will be released. The "service" subcommand controls idle timeouts on internal servers that are mapped into public addresses for external use. See the next page for a discussion of that variant.

Command Syntax

```
private idle [TCP|UDP|SERVICE] [timeout] {port}
```

Command Parameters

| Parameter Name | Description |
|----------------|--|
| TCP UDP | The protocol timer to be set (TCP or UDP). |
| SERVICE | Set idle timeout for inbound mapped services (TCP only). |
| timeout | The timeout in seconds for this protocol. The value entered will be rounded upward to the next 10-second multiple. |
| port | This may be used to specify a timeout value for a particular TCP or UDP service that is different from the global timeout value for that protocol. For example, you may wish to extend the idle timeout for telnet users beyond the global idle timeout, to reduce the number of times they are disconnected due to inactivity. The command "private idle tcp 7200 23" will set the idle timeout for the telnet service (service 23) to 7200 seconds (2 hours). This command variant may also be used to reduce the idle timeout for a specific TCP service. |

For Example

```
nscnat> private idle
Idle Timeouts: TCP: 3600 sec, UDP: 70 sec, TCP Ser: 3600 sec
nscnat> private idle udp 85
Idle Timeouts: TCP: 3600 sec, UDP: 90 sec, TCP Ser: 3600 sec
nscnat> private idle tcp 5400 23
Idle Timeouts: TCP: 3600 sec, UDP: 90 sec, TCP Ser: 3600 sec
TCP port 23: 5400 seconds, 0 uses
nscnat> _
```

You will note in the example, that "private idle" with no additional parameters will display the current settings. Disconnects caused by a per-port idle timeout are counted by service, with the number of disconnects displayed with the other information.

private idle service

Purpose

The private idle service sub-command differs from the TCP and UDP versions, in that the others set the idle timeout for internal clients accessing external servers, while this one sets the idle timeout for external clients accessing internal servers that have been mapped for external use.

Command Syntax

```
private idle SERVICE [timeout] {port}
```

Command Parameters

| Parameter Name | Description |
|----------------|--|
| SERVICE | Set idle timeout for inbound mapped services (TCP only). |
| timeout | The timeout in seconds for this protocol. The value entered will be rounded upward to the next 10-second multiple. |
| port | This may be used to specify a timeout value for a particular TCP service that is different from the global timeout value for all mapped servers. For example, you will wish to shorten the idle timeout for HTTP users to 30 seconds, and that of FTP users to 300 seconds. You may wish to lengthen the timeout for telnet users, if they are complaining about being disconnected prematurely. |

For Example

```
nscnat> private idle
Idle Timeouts: TCP: 3600 sec, UDP: 70 sec, TCP Ser: 3600 sec
nscnat> private idle tcp 30 80
Idle Timeouts: TCP: 3600 sec, UDP: 90 sec, TCP Ser: 3600 sec
TCP port 80: 30 seconds, 0 uses (service)
nscnat> private idle tcp 300 21
Idle Timeouts: TCP: 3600 sec, UDP: 90 sec, TCP Ser: 3600 sec
TCP port 21: 300 seconds, 0 uses (service)
TCP port 80: 30 seconds, 0 uses (service)
nscnat> _
```

You will note in the example, that “private idle” with no additional parameters will display the current settings. Disconnects caused by a per-port idle timeout are counted by service, with the number of disconnects displayed with the other information.

Note that only FTP needs a dynamic session when using port or default mapping. In Subnet Mode 3 mapping, FTP, telnet and traceroute get dynamic sessions.

ps

Purpose

The ps command displays the status of all processes running in the NetNAT.

Command Syntax

ps

Command Parameters

None.

For Example

```
nscnat> ps
ksigs 323327 queued 161329 hiwat 3 woken 322879 nops 4479 dups
309
kwaits 703145 nops 1014261 from int 0
PID      SP      stksize  event    fl  in  out  name
32a580   34a69c  32768   329fa0   IW          Cpanel listener
98fe8    200e6c  32768   99270    IW          en0 tx
229090   2491fc  32768   221140   IW          en1 tx
971b0    17904c  32768   54f5c    IW          network
98870    f9028   32768   0         I           display
96ff0    199058  32768   0         I           keyboard
98870    96a64   32768   0         I           cmdintrp
nscnat> _
```

On the first line of displayed output:

| | |
|--------|--|
| ksigs | Number of calls to ksignal to awaken a task. |
| queued | Ksignal calls that were queued because interrupts were masked. |
| hiwat | The deepest the queue has been since startup. |
| woken | Number of times processes have been awakened. |
| nops | Ksignal calls that didn't awaken anything. |
| dups | Ksignal calls that were duplicates. |

On the second line of displayed output:

| | |
|----------|--|
| kwaits | Number of calls to kwait to relinquish control or wait for an event. |
| nops | Kwait calls that didn't result in the calling process blocking. |
| from int | Kwait calls made from interrupt level (an error). |

In the individual process display section:

| | |
|---------|---|
| PID | The address of the entrypoint of this process. |
| SP | The address of the stack for this process. |
| stksize | The size of the stack for this process. |
| event | The address of the word used by ksignal and kwait to dispatch this process. |
| fl | Flags of I (interrupt state), W (waiting) and/or S (suspended). |
| in | File descriptor number for an input socket if any. |
| out | File descriptor number for an output socket if any. |
| name | The process' name. |

Additional Information

A process that is currently running will have an event of zero. Any time you display process status, the processes display, keyboard and cmdintrp will be running, since they are displaying your output. Other processes that seem always to be running may be looping. For example; If the "en1 rx" process is always running, then the receiver for ethernet 1 is probably in trouble in one way or another. If you are not experiencing operational problems, then the problem may not be a real problem. If you **are**, however, you may wish to make note of the conditions and symptoms and contact customer support for advice.

pwd

Purpose

The `pwd` command displays the path of the current working directory. This is just like the unix `pwd` command or the MS-DOS `cd` command (with no parameters). The working directory may be changed by use of the `cd` command.

Command Syntax

`pwd`

Command Parameters

None.

For Example

```
nscnat> pwd
c:/netnat
nscnat> _
```

rename

Purpose

The rename command renames a file in the NetNAT's DOS filesystem. This is just like the MS-DOS rename command.

Command Syntax

```
rename [old name] [new name]
```

Command Parameters

None other than the file names.

repeat

Purpose

The `repeat` command spawns a new console window and repeats a NetNAT command at intervals specified. This is very useful in watching interface or session activity, and memory utilization.

Command Syntax

```
repeat [interval] [NetNAT command with parameters]
```

Command Parameters

| Parameter Name | Description |
|----------------|---|
| interval | The command repeat interval expressed in milliseconds (1000 equals one second). |
| NetNAT command | The complete NetNAT command with any parameters required by that the command. |

For Example

```
repeat 1000 memory status
```

This command example will update a window displaying the current memory status every second. Toggle through the various console windows with the F8 function key. Terminate a repeat command session with control-C.

route

Purpose

The route IP routing command displays static information about each route known to the NetNAT (e.g., the IP Address of a destination host or network). The route sub-commands (see *Additional Information* below) allow you to add or delete the NetNAT's knowledge of a specific route.

Command Syntax

route

Command Parameters

None.

For Example

To view all the route information known by this NetNAT, the command is:

```
nscnat> route
Dest          Len Interface Gateway      Metric P Timer Use
10.1.1.255    32  en1
133.19.76.255 32  en0
10.1.1.0      24  en1
133.19.76.0   24  en0
10.0.0.0      8   en1      10.1.1.253  1   0   87311
default       0   en0      133.19.76.1  1   0   178334
nscnat> _
```

Displayed Information

| | |
|-----------|--|
| Dest | Destination network, host or broadcast address, or default. |
| Length | Length in bits of the netmask for this destination (24 bits is the same as the netmask 255.255.255.0). |
| Interface | The name of the interface to use to reach the specified destination. |
| Gateway | The IP Address of the next hop in the direction of the specified destination, if needed. |
| Metric | Whether "hops" are required to reach the destination. |
| P | "P" says that this route will not be published. |
| Timer | Time until this route expires (in seconds). |
| Use | Number of uses of this route. |

Additional Information

The route sub-commands inform the NetNAT about default and static routing information needed for your private network to gain access to internet and other remote hosts or networks. The route sub-commands are:

| | |
|-------------|---|
| add default | Sets the IP address and netmask for the NetNAT's default gateway to the Internet |
| add | Sets the IP address and netmask for routing traffic between the NetNAT and another specific private or public network. If the NetNAT is running a "routing protocol" process (like "RIP"), this route will be published to other computers on the LAN. |
| addprivate | Sets the IP address and netmask for routing traffic between the NetNAT and another specific private or public network. If the NetNAT is running a "routing protocol" process (like "RIP"), this route will not be published to other computers on the LAN. |
| drop | Removes a selected routing table entry. |
| flush | Removes any routing table entries that were learned by a dynamic routing protocol process. |
| lookup | Display the route that would be used by the NetNAT to attempt to reach a specified destination host address. |
| tunnel | Create or modify a route and define a form of tunnel to be used on that route. |

Described below are the command syntax and parameters for each route sub-command.

route add default

Purpose

The `route add default` sub-command sets the IP address and netmask for the gateway the NetNAT will use when routing traffic to the Internet or to any other network for which explicit routing information is not known.

Command Syntax

```
route add default [int name] [gateway]
```

Command Parameters

| | |
|-----------------------|--|
| <code>int name</code> | The name of a NetNAT logical interface on the default network (e.g., <code>en0</code>). |
| <code>gateway</code> | The IP Address of the default gateway on that network. |

| Parameter Name | Description |
|-----------------------|--|
| <code>int name</code> | The name of a NetNAT logical interface on the default network (e.g., <code>en0</code>). |
| <code>gateway</code> | The IP Address of the default gateway on that network. |

For Example

To set the default gateway to the internet on the NetNAT logical interface `en0` with an IP address of `204.79.22.15`, the command is:

```
route add default en0 204.79.22.15
```

Additional Information

The NetNAT uses the default route in the absence of other routing information. For any LAN segment that uses a router for connecting to the global internet, that router is the default route or gateway to the Internet for the entire LAN. All messages from the LAN that are addressed to the Internet will be directed to that router for subsequent delivery.

In most cases, the NetNAT is installed between your Internet access router and your internal network. You should set the NetNAT's default route to the IP Address of your Internet access router's ethernet port.

Since the NetNAT appears to your internal network as an Internet access router, the hosts and workstations on your LAN should be configured with the IP Address of the private ethernet port of the NetNAT, unless your network is additionally subnetted.

route add

Purpose

The route add sub-command sets the IP address and netmask for routing messages between your private local network and another specific private or public network. It also may enable one or more types of Virtual Private Network support.

Command Syntax

```
route add [dest net] [int name] [gateway]
```

Command Parameters

| Parameter Name | Description |
|----------------|---|
| dest net | The network IP Address/Netmask for a specific destination. Use NSC notation (network/#bits) as described in appendix. |
| int name | The name of the NetNAT logical interface to use when sending message to that destination. |
| gateway | The IP Address of the gateway that is the next hop in the direction of the specified destination. |

For Example

To create a static route to another network with the IP address 140.63.33 and a netmask of 255.255.255.0 (#bits = 24), using the NetNAT logical interface en0 and the gateway IP address 204.79.22.16, the command is:

```
route add 140.63.33/24 en0 204.79.22.16
```

Additional Information

A static route specifies a direct route to a given network. Without this explicit route information, the NetNAT will try to reach that network via the defined “default” route. For example, a network with an Internet connection and a dedicated link to an associated organization's network would have a “default” pointing to the Internet, and a static route (for the other network) pointing to the dedicated link.

route addprivate

Purpose

The route addprivate sub-command is exactly like the route add sub-command, except that the routing table entries created are never advertised by the NetNAT to other computers.

Command Syntax

```
route addprivate [dest net] [int name] [gateway]
```

Command Parameters

| Parameter Name | Description |
|----------------|---|
| dest net | The network IP Address/Netmask for a specific destination. Use NSC notation (network/#bits) as described in appendix. |
| int name | The name of the NetNAT logical interface to use when sending message to that destination. |
| gateway | The IP Address of the gateway that is the next hop in the direction of the specified destination. |

For Example

Please see examples from the previous section.

Additional Information

The route sub-command addprivate ensures that the NetNAT does not advertise this route if you start a routing protocol. Due to the complexity of configuring the routing protocols, we recommend that you DO NOT start one on the NetNAT. Feel free to run them on your other routers.

route drop

Purpose

The route drop sub-command causes the NetNAT to delete a specified routing table entry. The parameters you used to create the entry should be used on the drop sub-command, to make it possible for the NetNAT to locate the entry you wish to delete.

Command Syntax

```
route drop [dest net] [int name] [gateway]
```

Command Parameters

| Parameter Name | Description |
|----------------|---|
| dest net | The network IP Address/Netmask for a specific destination. Use NSC notation (network/#bits) as described in appendix. |
| int name | The name of the NetNAT logical interface to use when sending message to that destination. |
| gateway | The IP Address of the gateway that is the next hop in the direction of the specified destination. |

For Example

```
nscnat> route
Dest          Len Interface Gateway      Metric P  Timer  Use
10.1.1.255    32  en1          133.19.76.1    1      P  0      0
133.19.76.255 32  en0          133.19.76.1    1      P  0      0
10.1.1.0      24  en1          10.1.1.0       0      0    159622
133.19.76.0   24  en0          133.19.76.0    0      0    209331
10.0.0.0      8   en1          10.1.1.253    1      0    87311
default       0   en0          133.19.76.1    1      0    178334
nscnat> route drop 10.0.0.0/8 en1 10.1.1.253
nscnat> route
Dest          Len Interface Gateway      Metric P  Timer  Use
10.1.1.255    32  en1          133.19.76.1    1      P  0      0
133.19.76.255 32  en0          133.19.76.1    1      P  0      0
10.1.1.0      24  en1          10.1.1.0       0      0    159622
133.19.76.0   24  en0          133.19.76.0    0      0    209331
default       0   en0          133.19.76.1    1      0    178334
nscnat> _
```

route flush

Purpose

The route flush sub-command causes the NetNAT to delete any routes that were learned by way of a dynamic routing process (like RIP).

Command Syntax

route flush

Command Parameters

none

For Example

```
nscnat> route
Dest      Len Interface Gateway      Metric P  Timer  Use
10.1.1.255 32  en1          1          1    P  0      0
133.19.76.255 32 en0          1          1    P  0      0
10.1.1.0   24  en1          0          0      0  159622
133.19.76.0 24  en0          0          0      0  209331
10.0.0.0   8   en1          10.1.1.253 1      27   87311
default    0   en0          133.19.76.1 1      0    178334
nscnat> route flush
nscnat> route
Dest      Len Interface Gateway      Metric P  Timer  Use
10.1.1.255 32  en1          1          1    P  0      0
133.19.76.255 32 en0          1          1    P  0      0
10.1.1.0   24  en1          0          0      0  159622
133.19.76.0 24  en0          0          0      0  209331
default    0   en0          133.19.76.1 1      0    178334
nscnat> _
```

In this example, the route to the remainder of your intranet (the rest of 10.x.x.x) was initially “learned” by the NetNAT (as shown by a non-zero timer value). After the flush sub-command was executed, that route vanished from the routing table. If you are running a dynamic routing process on the NetNAT, the route will probably return shortly.

route lookup

Purpose

The route lookup sub-command results in a look-up of the specified host, and determination of the route that would be taken to attempt to access that host.

Command Syntax

```
route lookup [dest host]
```

Command Parameters

| Parameter Name | Description |
|----------------|--|
| dest host | The IP Address of a destination host or workstation. |

For Example

```
nscnat> route
Dest          Len Interface Gateway      Metric P  Timer  Use
10.1.1.255    32  en1          10.1.1.253    1    P  0      0
133.19.76.255 32  en0          133.19.76.1   1    P  0      0
10.1.1.0      24  en1          10.1.1.0      0    0  159622
133.19.76.0   24  en0          133.19.76.0   0    0      0
209331
10.0.0.0      8   en1          10.1.1.253    1    0  87311
default       0   en0          133.19.76.1   1    0  178334
nscnat> route lookup 138.113.17.112
default       0   en0          133.19.76.1   1    0  178334
nscnat> route lookup 10.2.99.6
10.0.0.0      8   en1          10.1.1.253    1    0  87311
nscnat> _
```

route tunnel

Purpose

The route tunnel sub-command creates an encapsulated Virtual Private Network (VPN) connection to a remote site. Messages from your local private LAN that are to go to the specified remote network will be encapsulated and sent via a tunnel to the other site. The type of encapsulation used on the tunnel depends on a "mode" parameter. The IP Addresses within the tunneled messages are not translated when this form of VPN is in effect.

Command Syntax

```
route tunnel [dest net] [int] [gateway] [mode] {spi}
```

Command Parameters

| Parameter Name | Description |
|----------------|---|
| dest net | The network IP Address/Netmask for a specific destination. Use NSC notation (network/#bits) as described in appendix. |
| int | The name of the NetNAT interface that should be used to reach the remote site. This may be an alias interface. |
| gateway | The IP Address of a compatible VPN gateway device (such as another NetNAT) corresponding to the other end of the VPN tunnel. |
| mode | The type of tunnel to be created. Examples are: I = RFC 2003 IP in IP Encapsulation M = RFC 2004 Minimal IP Encapsulation E = RFC 1827 Encapsulating Security Payload C = Enable Multi-Cast Traffic |
| spi | The Security Parameter Index when using mode e or a. This corresponds to a Security Association, which defines the encryption keys to be used. (See the command "secure add") |

For Example

To create an RFC 2003 IP-IP VPN tunnel to another network with the IP address 140.63.33 and a netmask of 255.255.255.0 (#bits = 24), using the compatible gateway IP address 204.79.22.16, the command is:

```
route tunnel 140.63.33/24 en0 204.79.22.16 i
```

Additional Information

The firewall features of the NetNAT prevent the acceptance of VPN traffic that has not been explicitly enabled by command. In addition, inbound Encapsulated VPN messages arriving from a known gateway containing packets from a network other than one specified by command are rejected by the NetNAT. To add more remote networks via the same gateway, use additional route tunnel commands.

route tunnel ... i

Purpose

The `route tunnel ... i` sub-command creates a IP-in-IP Encapsulation Virtual Private Network (VPN) connection to a remote site. Messages from your local private LAN that are to go to the specified remote network will be encapsulated and sent via an RFC 2003-compliant tunnel. The IP Addresses within the tunneled messages are not translated when this form of VPN is in effect.

Command Syntax

```
route tunnel [dest net] [int] [gateway] i
```

Command Parameters

| Parameter Name | Description |
|----------------|--|
| dest net | The network IP Address/Netmask for a specific destination. Use NSC notation (network/#bits) as described in appendix. |
| int | The name of the NetNAT interface that should be used to reach the gateway at the other end of the VPN tunnel. |
| gateway | The IP Address of a compatible VPN gateway device (such as another NetNAT) corresponding to the other end of the VPN tunnel. |
| i | Mode i (or I) is for IP-in-IP VPN technology. |

For Example

To create an RFC 2003 VPN tunnel to another network with the IP address 140.63.33 and a netmask of 255.255.255.0 (#bits = 24), using the compatible gateway IP address 204.79.22.16, the command is:

```
route tunnel 140.63.33/24 en0 204.79.22.16 i
```

Additional Information

The firewall features of the NetNAT prevent the acceptance of VPN traffic that has not been explicitly enabled by command. In addition, inbound IP-IP Encapsulation VPN messages from a known gateway containing packets from a network other than one specified by command are rejected by the NetNAT. To add more remote networks via the same gateway, use additional `route tunnel ... i` commands.

This mode of tunneling is compatible with Cisco's `tunnel mode nos`.

route tunnel ... m

Purpose

The `route tunnel ... m` sub-command creates an RFC 2004 Minimal Encapsulation Virtual Private Network (VPN) connection to a remote site. Messages from your local private LAN that are to go to the specified remote network will be encapsulated and sent via an RFC 2004-compliant tunnel. The IP Addresses within the tunneled messages are not translated when this form of VPN is in effect.

Command Syntax

```
route tunnel [dest net] [int] [gateway] m
```

Command Parameters

| Parameter Name | Description |
|----------------|--|
| dest net | The network IP Address/Netmask for a specific destination. Use NSC notation (network/#bits) as described in appendix. |
| int | The name of the NetNAT interface that should be used to reach the gateway at the other end of the VPN tunnel. |
| gateway | The IP Address of a compatible VPN gateway device (such as another NetNAT) corresponding to the other end of the VPN tunnel. |
| M | Mode M is for Minimal Encapsulation Protocol. |

For Example

To create an RFC 2004 Mincap VPN tunnel to another network with the IP address 140.63.33 and a netmask of 255.255.255.0 (#bits = 24), using the compatible gateway IP address 204.79.22.16, the command is:

```
route tunnel 140.63.33/24 en0 204.79.22.16 m
```

Additional Information

The firewall features of the NetNAT prevent the acceptance of VPN traffic that has not been explicitly enabled by command. In addition, inbound Minimal Encapsulation VPN messages from a known gateway containing packets from a network other than one specified by command are rejected by the NetNAT. To add more remote networks via the same gateway, use additional `route tunnel ... m` commands.

route tunnel ... e

Purpose

The `route tunnel ... e` sub-command creates an RFC 1827 Encapsulation Security Payload Virtual Private Network (VPN) connection to a remote site. Messages from your local private LAN that are to go to the specified remote network will be encapsulated and sent via an RFC 1825/1827-compliant tunnel. The IP Addresses within the tunneled messages are not translated when this form of VPN is in effect.

Command Syntax

```
route tunnel [dest net] [int] [gateway] e [spi]
```

Command Parameters

| Parameter Name | Description |
|-----------------------|---|
| <code>dest net</code> | The network IP Address/Netmask for a specific destination. Use NSC notation (network/#bits) as described in appendix. |
| <code>int</code> | The name of the NetNAT interface that should be used to reach the gateway at the other end of the VPN tunnel. |
| <code>gateway</code> | The IP Address of a compatible VPN gateway device (such as another NetNAT) corresponding to the other end of the VPN tunnel. |
| <code>e</code> | Mode <code>e</code> (or <code>E</code>) for the ESP Tunneling. |
| <code>spi</code> | Decimal number corresponding to the Security Parameters Index (see <code>secure add</code>) which defines the keys to use in ESP CBC-DES encryption and AH authentication. |

For Example

To create an RFC 1827 ESP VPN tunnel to another network with the IP address 140.63.33 and a netmask of 255.255.255.0 (#bits = 24), using the compatible gateway IP address 204.79.22.16 and the SPI of 1199, the command is:

```
route tunnel 140.63.33/24 en0 204.79.22.16 e 1199
```

Additional Information

The firewall features of the NetNAT prevent the acceptance of VPN traffic that has not been explicitly enabled by command. In addition, inbound ESP Encapsulation VPN messages from a known gateway containing packets from a network other than one specified by command are rejected by the NetNAT. To add more remote networks via the same gateway, use additional `route tunnel ... e` commands.

ESP tunneling is an integral part of IPSEC as preliminarily described in RFC 1825 and being discussed in IETF work groups.

route tunnel ... a

Purpose

The route tunnel ... a modifying sub-command adds an RFC 1826 Authentication Header to packets being tunneled via another tunneling mechanism. The result is the authentication of the sender, something that some of the tunneling protocols do not do. There is some level of authentication in the ESP tunnel, in that the originator would need to know the 56-bit DES key to be able to create valid messages. The "a" modifier may be added to any of the other tunneling modes.

Command Syntax

```
route tunnel [dest net] [int] [gateway] [e|i|m] a [spi]
```

Command Parameters

| Parameter Name | Description |
|----------------|--|
| dest net | The network IP Address/Netmask for a specific destination. Use NSC notation (network/#bits) as described in appendix. |
| int | The name of the NetNAT interface that should be used to reach the gateway at the other end of the VPN tunnel. |
| gateway | The IP Address of a compatible VPN gateway device (such as another NetNAT) corresponding to the other end of the VPN tunnel. |
| e or i or m | The mode of the tunneling that we are modifying to include an Authentication Header. AH works with any tunneling mode, but is most frequently found associated with the ESP mode. Together they constitute the IPSEC IP Security facility. |
| a | Mode modifier "a" adds an Authentication Header to the tunneled message. |
| spi | Decimal number corresponding to the Security Parameters Index (see secure add). |

For Example

To create an RFC 1827 ESP VPN tunnel to another network with the IP address 140.63.33 and a netmask of 255.255.255.0 (#bits = 24), using the compatible gateway IP address 204.79.22.16 and the SPI of 1199, and using an Authentication Header for originator authentication, the command is:

```
route tunnel 140.63.33/24 en0 204.79.22.16 e a 1199
```

Additional Information

The Authentication Header and ESP tunneling are integral parts of IPSEC as preliminarily described in RFC 1825 and being discussed in IETF work groups.

scrollback

Purpose

The scrollback command displays or modifies the size of the command window scrollback buffer (in lines kept). The default configuration retains the most-recent 1000 lines of command window for viewing in scroll mode (activated and deactivated by the F6 function key).

Command Syntax

scrollback {lines}

Command Parameters

| Parameter Name | Description |
|----------------|--|
| lines | Optional parameter specifying the number of lines to be retained by the console command window scroll feature. |

Additional Information

Omitting the “lines” parameter causes the scrollback command to display the current number of lines retained.

secure

Purpose

The various secure sub-commands display or modify information used in the creation of encrypted or authenticated services.

Command Syntax

```
secure [add|generate|status|trace]
```

Command Parameters

Depends on the subcommand.

For Example

To view all the Security Associations known by this NetNAT, the command is:

```
nscnat> secure status
SPI:      1199, currently 1 associations
  Dest          Len Interface  Gateway      Send pkts Recv
pkts
  10.1.1.255    32  en0(ae)      204.17.8.9
      1          0
nscnat> _
```

Displayed Information

| | |
|-----------|--|
| Dest | Destination network, host or broadcast address, or default. |
| Len | Length in bits of the netmask for this destination (24 bits is the same as the netmask 255.255.255.0). |
| Interface | The name of the interface to use to reach the specified destination. This includes the tunneling modes in parenthesis. |
| Gateway | The IP Address of the next hop in the direction of the specified destination, if needed. |
| Send pkts | Number of packets sent on this tunnel route. |
| Recv pkts | Number of packets received on this tunnel route. |

Additional Information

The secure sub-commands manage and display the Security Parameters Indices, the resulting Security Associations, and assist you in the selection of random DES keys. The secure sub-commands are:

| | |
|----------|--|
| add | Creates a Security Parameters Index of a specified number with a specified DES key. If the SPI already exists, the key is changed. |
| generate | Generates 56-bit DES keys using statistically un-correlated events and weak-key checking. |
| status | Displays the existing Security Parameters Indices and any related Security Association. |
| trace | Enables or disables some low-level debugging information. This is not interesting to the user. |

Described below are the command syntax and parameters for each secure sub-command.

secure add

Purpose

The secure add sub-command creates a Security Parameters Index (see RFC 1825 for details). An SPI contains security information that may be associated with a tunnel or other connection between computers or gateways. In the case of the NetNAT, this information consists of a 56-bit DES key to be used in encryption and authentication. When an SPI is applied to a tunnel between the NetNAT and a distant compatible gateway (such as another NetNAT), the resulting relationship is called a Security Association.

Command Syntax

```
secure add [spi] [key] {key 2}
```

Command Parameters

| Parameter Name | Description |
|----------------|---|
| spi | A decimal numeric value that uniquely defines this SPI. Numbers may range from 1 through $2^{32}-1$, with the numbers 1 through 255 reserved by the IANA for future use. |
| key | A 56-bit DES key represented as 16 hex digits (64 bits). Each byte of the key must have odd parity. In addition, there are about a hundred DES keys that are considered to be "weak" and the NetNAT will verify that you have not entered one of those. |
| key2 | An optional second 56-bit DES key represented as 16 hex digits (64 bits), as described above. This key is only needed when using Triple-DES encryption. |

For Example

To create an SPI numbered 1199, with a DES key of 498a269283df686e, the command is:

```
secure add 1199 498a269283df686e
```

Additional Information

The NetNAT will perform very basic key validation for you. The required odd parity will be validated, and then the key will be tested against the list of published "weak keys." Error messages of "Key needs odd parity" or "Illegal weak key" indicate problems with the entered key value. A simple facility to generate DES keys is provided by the secure generate sub-command. For exportable versions of the NetNAT encryption, the NetNAT will verify that the first 16 effective bits of the DES key are zero, thereby restricting the size of the key to 40 bits.

secure generate

Purpose

The secure generate sub-command uses the built-in NetNAT random number generation facility to suggest 56-bit DES keys suitable for use with the secure add sub-command (or with other non-NetNAT applications). The NetNAT uses a combination of RC4 encryption and the MD5 one-way message digest, continuously stimulated by a number of uncorrelated physical events, to produce random numbers with high entropy content. The secure generate sub-command takes 64 bits from this entropy pool, sets odd parity and tests the result against the list of weak keys. When a qualified result is found, it is displayed as a suggested key. For international NetNAT users, a 40-bit DES key is also generated and displayed. The 40-bit key is created by taking 64 bits from the entropy pool, zeroing the first 18 bits (each byte contains only 7 bits of meaningful information), setting odd parity and testing against the list of known weak keys. A 40-bit DES key will always start with 0101 (two bytes of zeros with odd parity). **No claim is made as to the appropriateness of any suggested key other than as described above.**

Command Syntax

secure generate

Command Parameters

None.

For Example

To generate a suggested DES key, the command is:

secure generate

A sample response would be:

Suggested domestic key: e0759ead8f9479c8

Suggested export key: 010126319e57fdc4

secure status

Purpose

The secure status sub-command displays the existing SPI entities and any Security Associations that exist. The keys for the defined SPI entities are never displayed.

Command Syntax

```
secure status
```

Command Parameters

None.

For Example

To display the current SPI and SA list, the command is:

```
nscnat> secure status
SPI:      1199, currently 2 associations
  Dest      Len Interface      Gateway      Send pkts Recv
pkts
  10.6.114.0 24 en0(e)      204.113.12.8 1742
1811
  10.6.200.0 24 en0(ai)     204.113.17.1 22374
21931
nscnat> _
```

Displayed Information

| | |
|-----------|--|
| Dest | Destination network or host. |
| Len | Length in bits of the netmask for this destination (24 bits is the same as the netmask 255.255.255.0). |
| Interface | The name of the interface to use to reach the specified destination. This includes the tunneling modes in parenthesis. |
| Gateway | The IP Address of the compatible security gateway at the other end of the secured tunnel. |
| Send pkts | Number of packets sent on this tunnel route. |
| Recv pkts | Number of packets received on this tunnel route. |

service mapping

The service mapping commands create the NetNAT mapping relationships between apparent external services and actual internal services, or controls mapping of internal user computers to apparent external IP Addresses when the standard “many-to-one” mapping is not desired. These mappings may be port-and-protocol conscious, or port independent. The service mapping sub-commands that create these mappings are:

| | |
|---------|--|
| Default | Maps all undefined service requests to an internal IP Address. |
| Fixed | Maps all services of an external apparent IP Address to an internal IP Address. |
| Local | Do not map this service, as the NetNAT processes this service locally. |
| Mux | Distributes incoming service requests across an array of servers. |
| Pool | Establishes a pool of apparent IP Addresses to be shared by internal users on a first-come-first-served basis. Using pool mapping, each internal user computer is assigned its own apparent IP Address as long as the computer remains active. Once the computer becomes idle, the apparent IP Address is released into the pool for another user computer. The pool is usually smaller than the total number of user computers. |
| Port | Maps a specific apparent IP Address/Port combination to an internal actual service. |
| Subnet | Maps an internal subnet to a single apparent address, or into a same-sized public subnet, depending on the specified mode. |
| Volume | Displays the traffic volumes for defined public services. |

In addition to the modes listed above, the NetNAT performs Dynamic Mapping of client IP Addresses into a single public IP Address, using port address translation algorithms developed by NSC in 1983. The NetNAT performs the mappings between "private" and "public" interfaces automatically. Therefore, no commands exist for this service.

Described below are the syntax and parameters for the service mapping sub-commands.

service ... default

Purpose

The service default sub-command sets the apparent and actual IP addresses for a server that will handle your network's undefined service requests.

Command Syntax

```
service [int name] default [app IP] [act IP] {flags}
```

Command Parameters

| Parameter Name | Description |
|----------------|--|
| int name | The name of the NetNAT logical interface on the public side (i.e., en0, tr0, ppp0) |
| app IP | The apparent IP Address for the public side of the logical interface. |
| act IP | The actual IP Address of the real server on your private network. |
| flags | An optional value that enables event monitoring and logging. E Enables logging of significant events that occur for the default Mapping service. See <i>Additional Information</i> for the default event log record format. V Enables logging of data volumes for default Mapping services. When set, the NetNAT will log counts of data segments and data bytes in and out every minute where the counts would be non-zero. |

For Example

To route an undefined service request received on the NetNAT's ethernet logical interface (e.g., en0 with apparent IP address 204.79.22.15), to the default Mapping server on your private network (e.g., actual IP address 192.168.200.5) and enable logging of default Mapping service events and volumes, the command is:

```
nscnat> service en0 default 204.79.22.15 192.168.200.5 E V
nscnat> service
en1 Main IP addr 192.168.200.254 No Mapping service
en0 Main IP addr 204.79.22.15 Default server 192.168.200.5:c000
    All private clients will use 204.79.22.15
nscnat> _
```

Additional Information

The `service default` sub-command establishes a last-resort default for responding to service requests that the NetNAT has not been able to handle otherwise. For example; If you suspect that Internet Bad Guys (IBGs) are attempting to invade your system, you may wish to set up a highly-instrumented host and point the NetNAT's Default Service Mapping filter at it. In this way, the NetNAT will not reject the connection attempts offhand, but will route them through to this instrumented internal machine for examination.

Because the NetNAT routes all undefined services to this default machine, you should carefully consider whether the use of the Default Service Mapping facility is wise. If you configure incorrectly and send the undefined service requests to an unprotected machine, you will compromise the NetNAT's Firewall functionality.

Any service request from the public area that is addressed to the apparent IP Address, and that is not otherwise handled by a Service Port Mapping relationship, is redirected to the configured service provider on the private network. Conversely, all reply messages from the actual IP Address (the actual server) are rewritten to appear to originate at the apparent public address.

All affected checksums are recalculated in the messages before they are transmitted.

service ... fixed

Purpose

The `service fixed` sub-command creates a relationship between all services and an apparent IP address on the public side of the NetNAT's logical interface, and the actual IP address for the real server on your private network.

This special mapping is useful when you have an internal server that processes many different services. Some users have experimental servers whose services are difficult to define, due to rapid changes. The fixed mapping afforded by this sub-command allows an administrator to open up the firewall for all services on that test machine. Needless to say, this may constitute a serious security risk if there are services on this test machine that may be compromised by an outsider.

Command Syntax

```
service [int name] fixed [app IP] [act IP] {flags}
```

Command Parameters

| Parameter Name | Description |
|----------------|--|
| int name | The name of the NetNAT logical interface on the public side (i.e., en0, tr0, ppp0) |
| app IP | The apparent IP Address for the public side of the logical interface. |
| act IP | The actual IP Address of the real server on your private network. |
| flags | An optional value that enables event monitoring and logging. E Enables logging of significant events that occur for the default Mapping service. See <i>Additional Information</i> for the default event log record format. V Enables logging of data volumes for default Mapping services. When set, the NetNAT will log counts of data segments and data bytes in and out every minute where the counts would be non-zero. |

For Example

```
nscnat> service en0 fixed 204.79.22.16 192.168.200.5
nscnat> service
en1 Main IP addr 192.168.200.254 No Mapping service
en0 Main IP addr 204.79.22.15
    All private clients will use 204.79.22.15
    204.79.22.16:(ALL) handled by 192.168.200.5::0
nscnat> _
```

Additional Information

As with the `service default` sub-command, the wisdom of using the `service fixed` facility must be weighed against the convenience it provides. While it allows the administrator to permit incoming service requests to new services without ongoing configuration involvement, it also permits access attempts on any service that the server computer is running. It may be appropriate to add filtering on your Internet access router to block incoming attempts at susceptible services.

service ... local

Purpose

The service local sub-command designates that the NetNAT, rather than a device on the private network, will process the specified service request. For example, HTTP or FTP, if permitted by the NetNAT.

Command Syntax

```
service [int] local [app IP] [app port] [TCP|UDP] {flags}
```

Command Parameters

| Parameter Name | Description |
|----------------|--|
| int name | The name of the NetNAT logical interface on the public side (i.e., en0, tr0, ppp0) |
| app IP | The apparent IP Address for the public side of the logical interface. |
| app port | The apparent Port Number for the NetNAT's service. |
| TCP UDP | The protocol to use when communicating with this NetNAT service, either TCP or UDP. |
| flags | An optional value that enables event monitoring and logging. E Enables logging of significant events that occur for the default Mapping service. See <i>Additional Information</i> for the default event log record format. V Enables logging of data volumes for default Mapping services. When set, the NetNAT will log counts of data segments and data bytes in and out every minute where the counts would be non-zero. |

For Example

```
nscnat> service en0 local 204.79.22.15 80 tcp
nscnat> service
en1 Main IP addr 192.168.200.254 No Mapping service
en0 Main IP addr 204.79.22.15
    All private clients will use 204.79.22.15
    204.79.22.15:80/TCP handled by local server
nscnat> _
```

Additional Information

When the NetNAT processes an inbound request for a service that has a "local Mapping" entry in the Mapping table, it checks for local listeners on the specified port. If there are none, the NetNAT rejects the service request.



service ... mux

Purpose

The service mux sub-command manages the mapping of service requests across a cluster of servers. The first invocation of the service mux command creates a virtual server that additional invocations augment. Up to 16 internal servers (located on the private side of the NetNAT) may share the responsibility for a single service. Inbound requests are distributed across the cluster based upon the number in queue for each server. This process is also called "load sharing" or "server leveling."

Command Syntax

```
service [int] mux [app IP] [app port] [TCP|UDP] [act IP]
[act port] {flags}
```

Command Parameters

| Parameter Name | Description |
|----------------|--|
| int | The name of the NetNAT logical interface on the public side (i.e., en0, tr0, ppp0) |
| app IP | The apparent IP Address for the public side of the logical interface. |
| app port | The apparent Port Number for the NetNAT's service. |
| TCP UDP | The protocol to use when communicating with this NetNAT service, either TCP or UDP. |
| act IP | The actual IP Address of an internal server to handle this service. |
| act port | The actual Port Number for the internal server's service daemon. |
| flags | An optional value that enables event monitoring and logging. E Enables logging of significant events that occur for the default Mapping service. See <i>Additional Information</i> for the default event log record format. V Enables logging of data volumes for default Mapping services. When set, the NetNAT will log counts of data segments and data bytes in and out every minute where the counts would be non-zero. |

For Example

To define an internal array of two Web servers to share requests addressed to the public address 204.79.22.15, use the following two commands:

```
nscnat> service en0 mux 204.79.22.15 80 tcp 192.168.200.3 80
nscnat> service en0 mux 204.79.22.15 80 tcp 192.168.200.4 80
nscnat> service
en1  Main IP addr 192.168.200.254 No Mapping service
en0  Main IP addr 204.79.22.15
     Mux Array: 204.79.22.15:80(TCP) served by:
       *192.168.200.3:80, 1278 uses, 11 now
       *192.168.200.4:80, 1209 uses, 13 now
     All private clients will use 204.79.22.15
nscnat> _
```

Additional Information

When the NetNAT receives the startup sequence for a service handled by a multiplex array, it looks at the current number of sessions existing to each of the internal servers, and sends the new connection to the least-busy server. At most sixteen servers may be sharing the service responsibility.

service ... pool

Purpose

The service pool sub-command establishes a pool of public IP Addresses that are assigned dynamically to internal clients on a one-to-one basis. A pool of addresses may be up to 512 addresses in length, and you may allocate as many as four pools.

Command Syntax

```
service [int name] pool [1st pub IP] [# in pool] {lease}
```

Command Parameters

| Parameter Name | Description |
|------------------------|--|
| int name | The name of the NetNAT logical interface on the public side (i.e., en0, tr0, ppp0) |
| 1 st pub IP | The apparent IP Address for the beginning of the pool. |
| # in pool | The number of contiguous public IP Addresses in the pool (maximum of 512 per pool). |
| lease | The length of time after an internal client goes idle (has no more sessions through the NetNAT) before the IP Address is released back into the pool. This parameter is expressed in tens of seconds (6 equals one minute), and is optional. If not provided, the lease time defaults to one day. In practice, setting this to a small value doesn't hurt anything, since all sessions have timed out. |

For Example

To establish a pool of 256 public IP Addresses beginning at 138.171.5.0, with a one-minute lease, use this command:

```
nscnat> service en0 pool 138.171.5.0 256 6
nscnat> service
en1 Main IP addr 192.168.200.254 No Mapping service
en0 Main IP addr 138.171.4.15
    Pool 1 at 138.171.5.0 with 256 members (0 in use), Lease 60 sec
    All other private clients will use 138.171.4.15
nscnat> _
```

Additional Information

The user command may be used to display the mapping at any given time.

service ... port

Purpose

The service port sub-command creates a relationship between the apparent IP address/Port Number/Protocol (UDP or TCP) combination for a service on the public side of the NetNAT and the actual IP Address/Port Number combination for the real server on your private network.

Command Syntax

```
service [int] port [app IP] [app port] [TCP|UDP] [act IP]
{act port} {flags}
```

Command Parameters

| Parameter Name | Description |
|----------------|---|
| int name | The name of the NetNAT logical interface on the public side (i.e., en0, tr0, ppp0) |
| app IP | The apparent IP Address for the public side of the logical interface. |
| app port | The apparent Port Number for the service. |
| TCP UDP | The protocol to use when communicating with this NetNAT service, either TCP or UDP. |
| act IP | The actual IP Address of the real server on your private network. |
| act port | The actual Port Number of the real service. If not specified, this defaults to the same number as the apparent Port Number. This field must be present if the optional "flags" field is specified. |
| flags | An optional value that enables event monitoring and logging. E Enables logging of significant events that occur for the default Mapping service. See <i>Additional Information</i> for the default event log record format. V Enables logging of data volumes for default Mapping services. When set, the NetNAT will log counts of data segments and data bytes in and out every minute where the counts would be non-zero. P This service is a PPTP Control Session. External PPTP clients will access an internal PPTP server by this service. At first access by the external client, a PPTP data object is created in the NetNAT for subsequent data traffic. Information about existing PPTP data objects may be displayed by use of the PPTP command. |

For Example

To permit the outside world to “see” HTTP, SMTP and DNS services on an apparent machine 204.79.22.15, when the actual services are in your private

Intranet on two machines 192.168.200.3 and 192.168.200.4 you would use these commands:

```
nscnat> service en0 port 204.79.22.15 80 tcp 192.168.200.3
80
nscnat> service en0 port 204.79.22.15 25 tcp 192.168.200.3
25
nscnat> service en0 port 204.79.22.15 53 udp 192.168.200.4
53
nscnat> service en0 port 204.79.22.15 53 tcp 192.168.200.4
53
nscnat> service
en1 Main IP addr 192.168.200.254 No Mapping service
en0 Main IP addr 204.79.22.15
    All private clients will use 204.79.22.15
    204.79.22.15:80/TCP   handled by 192.168.200.3:80
    204.79.22.15:25/TCP  handled by 192.168.200.3:25
    204.79.22.15:53/UDP  handled by 192.168.200.4:53
    204.79.22.15:53/TCP  handled by 192.168.200.4:53
nscnat> _
```

Additional Information

Any service request from the public area that is addressed to the apparent IP Address/Port Number combination is redirected to the actual service provider on the private network. Conversely, all reply messages from the actual IP Address/Port Number/Protocol combination are rewritten to appear to originate at the apparent public combination.

service ... subnet

Purpose

The service subnet sub-command is the most complex of the service commands, and is the most powerful. This command defines unique treatment for all or part of your internal network, based upon subnet. The size of the subnet may be anything that is possible to define with a non-comb netmask, from a single computer (with a 32-bit mask or 255.255.255.255) to an entire Class-A network, or even larger.

Command Syntax

```
service [int name] subnet [pub IP] [pri net] [mask] 0
service [int name] subnet [pub net] [pri net] [mask] 1
service [int name] subnet [pub net] [pri net] [mask] 2
service [int name] subnet [pub net] [pri net] [mask] 3
```

Command Parameters

| Parameter Name | Description |
|----------------|--|
| int name | The name of the NetNAT logical interface on the public side (i.e., en0, tr0, ppp0) |
| pub IP | The public (apparent) IP Address to be shared by all members of the subnet. |
| pri net | The subnet address for the internal subnet we're defining. |
| mask | The netmask for both the public subnet and for the internal subnet we're defining. |
| pub net | The subnet address for the public subnet we're defining. |
| mode | Mode 0, 1, 2 or 3. This defines the characteristics and special treatment of members of this subnet. If not specified, mode 0 is used. |

For Example

Examples will be given in the subsequent sections detailing the four modes.

service ... subnet mode 0

Purpose

The service subnet mode 0 mapping forces all members of the described subnet to share a single public IP Address. The computers may share a single address because of our port address translation facility. Unless enabled by a service port command, no inbound service attempts will be permitted through the NetNAT into the described subnet. Inbound ping messages will be replied to by the NetNAT itself.

Command Syntax

```
service [int name] subnet [pub IP] [pri net] [mask] 0
```

Command Parameters

| Parameter Name | Description |
|----------------|--|
| int name | The name of the NetNAT logical interface on the public side (i.e., en0, tr0, ppp0) |
| pub IP | The public (apparent) IP Address to be shared by all members of the subnet. |
| pri net | The subnet address for the internal subnet we're defining. |
| mask | The netmask for the internal subnet we're defining. |
| mode | Zero (0) to specify Mode 0. |

For Example

To cause members of internal subnet 10.1.2.0/24 to share public IP Address 204.79.3.6, use this command:

```
nscnat> service en0 subnet 204.79.3.6 10.1.2.0 255.255.255.0
0
nscnat> service
en1 Main IP addr 10.1.1.254 No Mapping service
en0 Main IP addr 204.79.3.15
      Net 10.1.2.0 mask 255.255.255.0 will use 204.79.3.6
      All other private clients will use 204.79.3.15
nscnat> _
```

Additional Information

It is perfectly legal to have several subnet mapping definitions, even ones that overlap. For example, you could define a mode 0 setup like in the illustration above, and then define an even more restrictive one on top of it. We define “more restrictive” to mean that the netmask masks the network down to a smaller net. You could even define a single computer as a subnet by using 255.255.255.255 as the netmask. In this way, you may have a whole Class-C network share one public IP Address, and give a second public IP Address to a single computer in that network. The look-up algorithm will always find the “most restrictive” subnet

definition first, so it doesn't matter in what order your configuration entries are entered.

service ... subnet mode 1

Purpose

The service subnet mode 1 performs what we call a "block translation," where an internal subnet is mapped as a block into a same-sized public subnet. In this way, each internal computer has a predictable public address for Internet operations. All firewall functions are in place in mode 1, so no inbound service attempts will be permitted through the NetNAT into the internal subnet. Inbound ping messages will be replied to by the NetNAT itself, making the subnet appear to be completely full of computers.

Command Syntax

```
service [int name] subnet [pub net] [pri net] [mask] 1
```

Command Parameters

| Parameter Name | Description |
|----------------|--|
| int name | The name of the NetNAT logical interface on the public side (i.e., en0, tr0, ppp0) |
| pub net | The subnet address for the public subnet we're defining. |
| pri net | The subnet address for the internal subnet we're defining. |
| mask | The netmask for both the public subnet and for the internal subnet we're defining. |
| mode | One (1) to specify Mode 1. |

For Example

To cause members of internal subnet 10.1.2.0/24 to appear to be in public IP subnet 204.79.3.0/24, use this command:

```
nscnat> service en0 subnet 204.79.3.0 10.1.2.0 255.255.255.0 1
nscnat> service
en1 Main IP addr 10.1.1.254 No mapping service
en0 Main IP addr 204.79.3.15
    Net 10.1.2.0 mask 255.255.255.0 will use 204.79.3.0, mode 1
    All other private clients will use 204.79.3.15
nscnat> _
```

Additional Information

As mentioned in the mode 0 section, subnet mapping definitions may be combined without problems. You may use a mode 1 mapping to place all members of an internal subnet into a public subnet, and then overlay it with a more-restrictive mode 0 definition to give a subset of the subnet a completely different shared public address.

service ... subnet mode 2

Purpose

The service subnet mode 2 is another “block translation” mechanism, and is similar to mode 1, but with a natural extension. As before, each internal computer has a predictable public address for Internet operations, but in mode 2, pings are passed into the internal subnet for the actual computer to handle. This lets you verify connectivity through the NAT, but without opening your internal network to attacks from without. All other firewall functions are in place in mode 2, so no other inbound service attempts will be permitted through the NetNAT into the internal subnet.

Command Syntax

```
service [int name] subnet [pub net] [pri net] [mask] 2
```

Command Parameters

| Parameter Name | Description |
|----------------|--|
| int name | The name of the NetNAT logical interface on the public side (i.e., en0, tr0, ppp0) |
| pub net | The subnet address for the public subnet we're defining. |
| pri net | The subnet address for the internal subnet we're defining. |
| mask | The netmask for both the public subnet and for the internal subnet we're defining. |
| mode | Two (2) to specify Mode 2. |

For Example

To cause members of internal subnet 10.1.2.0/24 to appear to be in public IP subnet 204.79.3.0/24, use this command:

```
nscnat> service en0 subnet 204.79.3.0 10.1.2.0 255.255.255.0 2
nscnat> service
en1 Main IP addr 10.1.1.254 No mapping service
en0 Main IP addr 204.79.3.15
Net 10.1.2.0 mask 255.255.255.0 will use 204.79.3.0, mode 2
All other private clients will use 204.79.3.15
nscnat> _
```

Additional Information

As mentioned before, subnet mapping definitions may be combined without problems.

service ... subnet mode 3

Purpose

The service subnet mode 3 is the newest of our “block translation” modes, adding SNMP payload translation (and optional telnet/traceroute passthrough) to the ping passthrough feature of mode 2. This combination allows network discovery and management of a private network from the public side of the NetNAT. All other firewall functions are in place in mode 3, so no other inbound service attempts will be permitted through the NetNAT into the internal subnet.

Command Syntax

```
service [int] subnet [pub net] [pri net] [mask] 3 {flags}
```

Command Parameters

| Parameter Name | Description |
|----------------|--|
| int | The name of the NetNAT logical interface on the public side (i.e., en0, tr0, ppp0) |
| pub net | The subnet address for the public subnet we're defining. |
| pri net | The subnet address for the internal subnet we're defining. |
| mask | The netmask for both the public subnet and for the internal subnet we're defining. |
| mode | Three (3) to specify Mode 3. |
| flags | "T", "F" and/or "R" to permit inbound telnet, ftp and/or traceroute. This is optional. |

For Example

To cause members of internal subnet 10.0.0.0/8 to appear to be in public IP subnet 12.0.0.0/8, and permit network discovery and management via mode 3 use this command:

```
nscnat> service en0 subnet 12.0.0.0 10.0.0.0 255.0.0.0 3
nscnat> service
en1 Main IP addr 10.1.1.254 No mapping service
en0 Main IP addr 204.79.3.15
Net 10.0.0.0 mask 255.0.0.0 will use 12.0.0.0, mode 3
All other private clients will use 204.79.3.15
nscnat> _
```

Additional Information

The rules that control the translation of SNMP payloads are kept in a user-editable file in the NetNAT runtime directory, permitting you to add new translation rules as you detect MIB objects that are being passed through without modification.

service ... subnet mode 3 t

Purpose

Service subnet mode 3 may be modified with the "T" suffix, to add inbound telnet session passthrough to the SNMP payload translation and ping passthrough features of mode 3. This permits terminal session connectivity to the managed devices on the private side of the NetNAT. All other firewall functions are in place in mode 3, so no other inbound service attempts will be permitted through the NetNAT into the internal subnet.

Command Syntax

```
service [int] subnet [pub net] [pri net] [mask] 3 t
```

Command Parameters

| Parameter Name | Description |
|----------------|--|
| int | The name of the NetNAT logical interface on the public side (i.e., en0, tr0, ppp0) |
| pub net | The subnet address for the public subnet we're defining. |
| pri net | The subnet address for the internal subnet we're defining. |
| mask | The netmask for both the public subnet and for the internal subnet we're defining. |
| mode | Three (3) to specify Mode 3. |
| t | This flag adds inbound telnet session passthrough to the other features of Mode 3. The "t" flag may be combined with others. |

For Example

To cause members of internal subnet 10.0.0.0/8 to appear to be in public IP subnet 12.0.0.0/8, permitting network discovery and both telnet and SNMP management via mode 3 use this command:

```
nscnat> service en0 subnet 12.0.0.0 10.0.0.0 255.0.0.0 3 t
nscnat> service
en1 Main IP addr 10.1.1.254 No mapping service
en0 Main IP addr 204.79.3.15
    Net 10.0.0.0 mask 255.0.0.0 will use 12.0.0.0, mode 3:t
    All other private clients will use 204.79.3.15
nscnat> _
```

Additional Information

Permitting inbound telnet to a network is a security risk, and this should be given thoughtful consideration before enabling this feature.

service ... subnet mode 3 f

Purpose

Service subnet mode 3 may be modified with the "F" suffix, to add inbound FTP session passthrough to the SNMP payload translation and ping passthrough features of mode 3. This permits file transfer session connectivity to the managed devices on the private side of the NetNAT. All other firewall functions are in place in mode 3, so no other inbound service attempts will be permitted through the NetNAT into the internal subnet, unless otherwise permitted by NetNAT configuration.

Command Syntax

```
service [int] subnet [pub net] [pri net] [mask] 3 f
```

Command Parameters

| Parameter Name | Description |
|----------------|---|
| int | The name of the NetNAT logical interface on the public side (i.e., en0, tr0, ppp0) |
| pub net | The subnet address for the public subnet we're defining. |
| pri net | The subnet address for the internal subnet we're defining. |
| mask | The netmask for both the public subnet and for the internal subnet we're defining. |
| mode | Three (3) to specify Mode 3. |
| f | This flag adds inbound FTP session passthrough to the other features of Mode 3. The "f" flag may be combined with others. |

For Example

To cause members of internal subnet 10.0.0.0/8 to appear to be in public IP subnet 12.0.0.0/8, permitting network discovery and ftp, telnet, and SNMP management via mode 3 use this command:

```
nscnat> service en0 subnet 12.0.0.0 10.0.0.0 255.0.0.0 3 t f
nscnat> service
en1 Main IP addr 10.1.1.254 No mapping service
en0 Main IP addr 204.79.3.15
Net 10.0.0.0 mask 255.0.0.0 will use 12.0.0.0, mode
3:tf
All other private clients will use 204.79.3.15
nscnat> _
```

Additional Information

Permitting inbound FTP to a network is a security risk, and this should be given thoughtful consideration before enabling this feature. Please see the "private idle service" subcommand for important additional information.

service ... subnet mode 3 r

Purpose

Service subnet mode 3 may be modified with the "R" suffix, to add inbound traceroute passthrough to the SNMP payload translation and ping passthrough features of mode 3. This enables the use of the powerful traceroute tool by the professional network manager. All other firewall functions are in place in mode 3, so no other inbound service attempts will be permitted through the NetNAT into the internal subnet, unless otherwise permitted by NetNAT configuration.

Command Syntax

```
service [int] subnet [pub net] [pri net] [mask] 3 r
```

Command Parameters

| Parameter Name | Description |
|----------------|--|
| int | The name of the NetNAT logical interface on the public side (i.e., en0, tr0, ppp0) |
| pub net | The subnet address for the public subnet we're defining. |
| pri net | The subnet address for the internal subnet we're defining. |
| mask | The netmask for both the public subnet and for the internal subnet we're defining. |
| mode | Three (3) to specify Mode 3. |
| r | This flag adds inbound traceroute passthrough to the other features of Mode 3. The "r" flag may be combined with others. |

For Example

To cause members of internal subnet 10.0.0.0/8 to appear to be in public IP subnet 12.0.0.0/8, permitting network discovery and traceroute, telnet, and SNMP management via mode 3 use this command:

```
nscnat> service en0 subnet 12.0.0.0 10.0.0.0 255.0.0.0 3 t r
nscnat> service
en1 Main IP addr 10.1.1.254 No mapping service
en0 Main IP addr 204.79.3.15
Net 10.0.0.0 mask 255.0.0.0 will use 12.0.0.0, mode
3:tr
All other private clients will use 204.79.3.15
nscnat> _
```

Additional Information

Like the other Mode 3 Modifiers, the traceroute option may be combined with others, to provide the services needed.

service ... volume

Purpose

The service volume command displays additional traffic volume information about the services mapped using the various service mapping commands.

Command Syntax

```
service [int name] volume
```

Command Parameters

| Parameter Name | Description |
|----------------|--|
| int name | The name of the NetNAT logical interface on the public side (i.e., en0, tr0, ppp0) |

For Example

```
nscnat> service en0 volume
Service volumes for en0
 204.79.3.15:80/TCP ( 10.1.1.30:80)
  p_in 2206, p_out 3112, b_in 23622, b_out 119835, b/s_in 56, b/s_out
3612
nscnat> _
```

Additional Information

In this example, a public web server's traffic counts are displayed. The last two values are bytes per second in and out. If you are not using syslog to log volumes, these will be a long-term average bytes per second. If you **are** using syslog to log volumes, the seconds-since-definition value that this command uses to compute the long-term average is used otherwise, resulting in inaccurate values displayed.

Since the displayed bytes per second is computed over lots of time, a lot of volume is required before non-zero values are displayed.

source(b)

Purpose

The source command reads NetNAT commands from a script file and executes them. The source command executes them in the foreground, and the sourceb command executes them in the background. The difference between the two is largely that sourceb is run in a spawned session, whereas source is run in the main console session. Source is very convenient for tasks like deleting old packet trace files and starting a fresh packet trace.

Command Syntax

```
source [file name]  
sourceb [file name]
```

Command Parameters

| Parameter Name | Description |
|----------------|--|
| file name | The name of a file containing NetNAT commands. |

syslog

Purpose

The syslog command displays the current logging status for the NetNAT, disables logging, and sets or changes the IP address of the syslog server. Information includes the IP Address of the configured syslog host and the state of the message reject logging variable, if it is non-zero.

Command Syntax

```
syslog [ipaddress|hostname]
```

Command Parameters

| Parameter Name | Description |
|-----------------------|--|
| blank | Displays the current logging status for the NetNAT. |
| 0 | When present, zero (0) disables the syslog server. |
| IP Address Hostname | Sets or changes the IP address or hostname (if the NetNAT supports DNS) for the syslog server. |

For Example

```
syslog
```

Possible responses are:

"No syslog server" when no server has been specified.

"Syslog server is xx.xx.xx.xx, using facility: local2, priority: info" when server xx.xx.xx.xx is in use.

If reject logging is enabled, that information will be displayed, too.

Additional Information

Syslog originated in the Unix world and provides a means of logging significant information to a remote server for processing. The NetNAT uses a syslog server to collect useful event information and data volume counts. This eliminates the need to locally store and process statistical data. Computers that have more time on their hands are better at distilling and graphing statistical information.

syslog facility

Purpose

The syslog facility command changes the facility code under which the NetNAT logs all events. The default facility is "LOCAL2," but may be changed by means of this command.

Command Syntax

```
syslog facility [facility]
```

Command Parameters

| Parameter Name | Description |
|----------------|---|
| facility | Specifies the syslog facility to use for all syslog messages. The default value is "local2." Legal values are kern, user, mail, daemon, auth, syslog, lpr, news, uucp, cron, authpriv, local0, local1, local2, local3, local4, local5, local6 and local7. |

For Example

To change the syslog facility that the NetNAT uses for all syslog messages to "daemon" from the default "local2," the command is:

```
syslog facility daemon
```

There is no response when the command syntax is correct.

syslog priority

Purpose

The syslog priority command changes the priority code under which the NetNAT logs all non-reject events. The default priority is "INFO," but may be changed by means of this command. Reject events are changed with the "syslog reject" command.

Command Syntax

```
syslog priority [priority]
```

Command Parameters

| Parameter Name | Description |
|----------------|---|
| priority | Optional parameter specifying the syslog priority to use for all syslog messages other than reject messages (possible values are alert, crit, debug, emerg, err, info, notice and warning. The default value is "info." |

For Example

To change the syslog priority that the NetNAT uses for all non-reject syslog messages to "notice" from the default "info," the command is:

```
syslog priority notice
```

There is no response when the command syntax is correct.

syslog reject

Purpose

The syslog reject command enables or disables logging, via syslog, of pertinent information about service requests that the NetNAT rejects.

Command Syntax

```
syslog reject [on|off|1|0|true|false] {priority}
```

Command Parameters

| Parameter Name | Description |
|-----------------|--|
| off false 0 | Disables the NetNAT from sending the syslog server log records for rejected service request events. |
| on true 1 | Enables the NetNAT to send the syslog server log records for rejected service request events. |
| priority | Optional parameter specifying the syslog priority to use for the reject messages (possible values are alert, crit, debug, emerg, err, info, notice and warning. The default value is "info." |

For Example

To disable the NetNAT from routing log records for rejected service requests to the syslog server, the command is:

```
syslog reject off
```

The response is "Reject logging disabled."

To instruct the NetNAT to log records for rejected service requests using syslog priority "notice" rather than the default "info," the command is:

```
syslog reject 1 notice
```

Additional Information

Message rejection is a global feature of the NetNAT and logging control is centered in the syslog server facility. When enabled, the NetNAT will send a log record for each rejected service request to the syslog server for logging.

Give careful thought before enabling reject logging, as it can produce a substantial amount of data.

tcp

Purpose

The tcp command displays the contents of the TCP “MIB” (the SNMP Management Information Block containing the TCP statistics), as well as any sockets with listeners and connections.

Command Syntax

tcp status

Command Parameters

None.

```
nscnat> tcp status
( 1)tcpRtoAlgorithm      4 ( 2)tcpRtoMin          0
( 3)tcpRtoMax           94967295 ( 4)tcpMaxConn        4294967295
( 5)tcpActiveOpens      2 ( 6)tcpPassiveOpens    1
( 7)tcpAttemptFails     2 ( 8)tcpEstabResets     0
( 9)tcpCurrEstab        1 (10)tcpInSegs         60
(11)tcpOutSegs          57 (12)tcpRetransSegs     1
(14)tcpInErrs           0 (15)tcpOutRsts        8
   &TCB Rcv-Q Snd-Q      Local socket Remote socket State
19e960   0   600      10.1.1.254:23 10.1.1.6:1979 Estab
19eaf8   0   0         0.0.0.0:23   0.0.0.0:0 Listen (S)
34a5b8   0   0         0.0.0.0:113  0.0.0.0:0 Listen (S)
32a220   0   0         0.0.0.0:21   0.0.0.0:0 Listen (S)
32a0b8   0   0         0.0.0.0:80   0.0.0.0:0 Listen (S)
329f70   0   0         0.0.0.0:9    0.0.0.0:0 Listen (S)
329e70   0   0         0.0.0.0:7    0.0.0.0:0 Listen (S)
nscnat> _
```

trace

Purpose

The trace command enables or disables packet capture for the interfaces of the NetNAT. Packets may be captured to the console display or to file for later viewing. Depending upon the flags that are entered to start the capture, the traced data will be more or less decoded for ease of human consumption. Check out ifconfig tracelimit for a nice facility to keep trace files from filling your entire disk.

Command Syntax

```
trace
trace [int name]
trace [int name] off
trace [int name] [flags]
trace [int name] [flags] [filename]
```

The first form displays whether tracing is occurring on any of the NetNAT interfaces. The second form displays the status of tracing for a particular interface. The third form terminates tracing on a specified interface, closing any output file that may have been in use.

The last two forms start the tracing process. The flags determine what is captured and how it is interpreted before display or filing. With the “filename” parameter, the output goes to a file for subsequent viewing.

Command Parameters

| Parameter Name | Description |
|----------------|--|
| int name | The name of the NetNAT logical interface (e.g. en0). |
| flags | A numeric value representing bit settings to control what gets captured and how the captured data will be interpreted for display. |

Additional Information

The flag bit settings are as follows:

| Bits value in Hex | Description |
|-------------------|---|
| 001 | Trace output packets. |
| 010 | Trace input packets. |
| 100 | Dump packets in ASCII (hex is the default). |
| 200 | Dump packets in hex and ASCII both. |
| 1000 | Suppress capture of broadcast messages. |

The flag bit values to be combined to achieve the desired characteristics. For example, our favorite combination is “211” which causes both input and output packets to be captured and displayed in hex and ASCII.

udp

Purpose

The `udp` command displays the contents of the UDP “MIB” (the SNMP Management Information Block containing the UDP statistics), as well as any sockets with listening processes.

Command Syntax

udp status

Command Parameters

None.

Possible responses are:

- ◆ Display of UDP statistics.

```
nscnat> udp status
( 1)udpInDatagrams      6      ( 2)udpNoPorts        229
( 3)udpInErrors         0      ( 4)udpOutDatagrams   37
      &UCB  Rcv-Q  Local socket
000269200      0    0.0.0.0:1649
nscnat> _
```

users

Purpose

The `users` command works on NetNATs that have limited-user licenses. It shows the number of permitted simultaneous users, the current number active, and the IP Addresses of the active users and the number of sessions each has in use. The `user pop` sub-command enables and controls the artificial-intelligence predictive dialer (available in DialNAT models only). The `user pop` sub-command will be detailed in subsequent sections.

Command Syntax

```
users  
users pop
```

Command Parameters

None.

For Example

Typical responses are:

For a NetNAT with an unlimited-user license:

```
nscnat> users  
Unlimited mode... No client records kept.  
nscnat> _
```

For a NetNAT with an 8-user license:

```
nscnat> users  
8 Client mode, 2 in use.  
Client 1 at address 10.1.1.6, 1 sessions  
Client 2 at address 10.1.1.30, 3 sessions  
nscnat> _
```

version

Purpose

The version command displays the NetNAT version number, the serial number, and time since the NetNAT program started.

Command Syntax

| | |
|---------|---|
| version | Display the NetNAT version, serial number and uptime. |
|---------|---|

For Example

view

Purpose

View is a file pager, and is in fact an alias for the more command described earlier. View creates a new console window that shows the contents of the requested file, and allows you to scroll up and down through it. To exit the view window, type q or control-C.

Command Syntax

| | |
|------------------------------|---------------------------------------|
| <code>view [filename]</code> | Display a file in a scrolling window. |
|------------------------------|---------------------------------------|

Additional Information

The normal PC PgUp, PgDn, Home, End, and up-down arrow keys let you scroll around in the specified file. Dedicated emacs and vi addicts will be pleased to discover that their favorite up-down line keys (and a few others) also work with the view command.

watchdog

Purpose

The watchdog global command sets the timeout variable for the built-in hardware watchdog timer.

Command Syntax

```
start watchdog [parm]
```

Command Parameters

| Parameter | Timeout in seconds |
|-----------|--------------------|
| 0 | Watchdog OFF |
| 1 | 0.5 |
| 3 | 2.1 |
| 5 | 8.6 |
| 7 | 17.2 |
| 9 | 34.4 |
| 11 | 137.9 |
| 13 | 275.8 |
| 15 | 551.6 |

For Example

To set the watchdog timer so that it resets the hardware within 8.6 seconds after a program crash or hardware hang, the command is:

```
start watchdog 5
```

To set the watchdog timer to the default reset time period (17.2), the command is:

```
start watchdog
```

There is no response after setting the reset time for the watchdog timer.

Additional Information

The watchdog timer is an independent device inside the NetNAT that monitors the NetNAT's operations. In the event of a program crash or hardware hang, the watchdog waits the indicated time-out period, then resets the NetNAT hardware, rebooting the NetNAT PC.

We recommend setting the watchdog time to either a value of 5 (for 8.6 seconds), or the default value of 7 (for 17.2 seconds). A higher setting could result in unnecessary delays when restarting the unit.

WWW

Purpose

The www global command starts the World Wide Web server.

Command Syntax

```
start www [port]
```

Command Parameters

| Parameter Name | Description |
|----------------|--|
| port | Indicates the port to use for the web server. Set this parameter only if your web server uses some other port than the default. When blank, the NetNAT uses the standard WWW port of 80. |

For Example

To start the web server using the default port value of 80, the command is:

```
start www
```

To start the web server using port 60, the command is:

```
start www 60
```

Additional Information

This page intentionally left blank