

# Configuring IOS NAT

Michael J. Martin

During the next few months, we will be examining solutions utilizing Cisco's Firewall and Intrusion Detection IOS (FW/IDS IOS). Introduced in the 12.x IOS tree, the FW/IDS IOS represents a shift in Cisco's IOS from "pure routing" to a convergence of data, voice routing and application level network services.

This month we will take a look at Network Address Translation (NAT), which performs the function of altering the IP network (OSI-L3) layer and transport protocol (OSI-L4) addressing information contained in the headers and data portion of network packets. NAT was designed to provide Internet access to a stub network using non/unroutable addressing and later expanded to provide many-to-one connection "sharing" services. The original motivation for NAT is stated clearly in the abstract of RFC 1631: "It is possible that CIDR (Classless Interdomain Routing) will not be adequate to maintain the IP Internet until long-term solutions are in place. This memo proposes another short-term solution, address reuse, that complements CIDR or even makes it unnecessary... to place Network Address Translators at the borders of stub domains." NAT, like CIDR, was intended to address an early fear in the commercialization of the Internet, which was the depletion of the IPv4 address space before a new addressing scheme could be agreed upon and adopted.

As it turns out, CIDR has been very successful, the IP "address crisis" was averted, and IPv6 has still not been adopted (yet) for addressing the Internet. Nonetheless, NAT's advocates and early implementers had a very solid idea that has become a very effective tool for addressing certain types of network scaling issues. Today, the most common use for NAT is implementing "connection sharing" on small and home office broadband gateway solutions (i.e., hardware-based broadband routers from companies such as Linksys and Netgear and software solutions like Microsoft's interconnection sharing option and WinGate). Some other uses of NAT are:

- Transition services for IP network conversions
- Unidirectional "network-in-network" secure wireless LAN and LAN enclaves.
- Providing high availability/server load balancing (HA/SLB) service
- IPv4 to IPv6 protocol translation (defined in RFC 2766 and not covered in this article)

While obscured, NAT is also a key component of most modern firewall implementations. This is largely due to similarities between NAT and the stateful connection tracking in terms of operational modality and complementary functionality. It is from this context that we will be examining the Cisco IOS NAT implementation. Readers should bear in mind that NAT is not a dependent technology for firewall implementations. However, the proliferation of inexpensive broadband access technologies, which often require "connection sharing" for an effective implementation, make the use of NAT (and a understanding of how it works) more of a design requirement than an option.

## What is Network Address Translation?

The problem with understanding and implementing NAT is not grasping the concept, but rather understanding the jargon. NAT is defined in RFC 1631 (the basis for Cisco's NAT implementation) and refined in RFC 2663. The definition of NAT above defines the action performed by NAT. We also need to take a look at the function of NAT. To do so requires us to define some terms:

- **Network domain:** An IP network or networks associated under a common routing policy (a concept similar to that of an Autonomous System (AS) but without the administrative relation

## Configuring IOS NAT

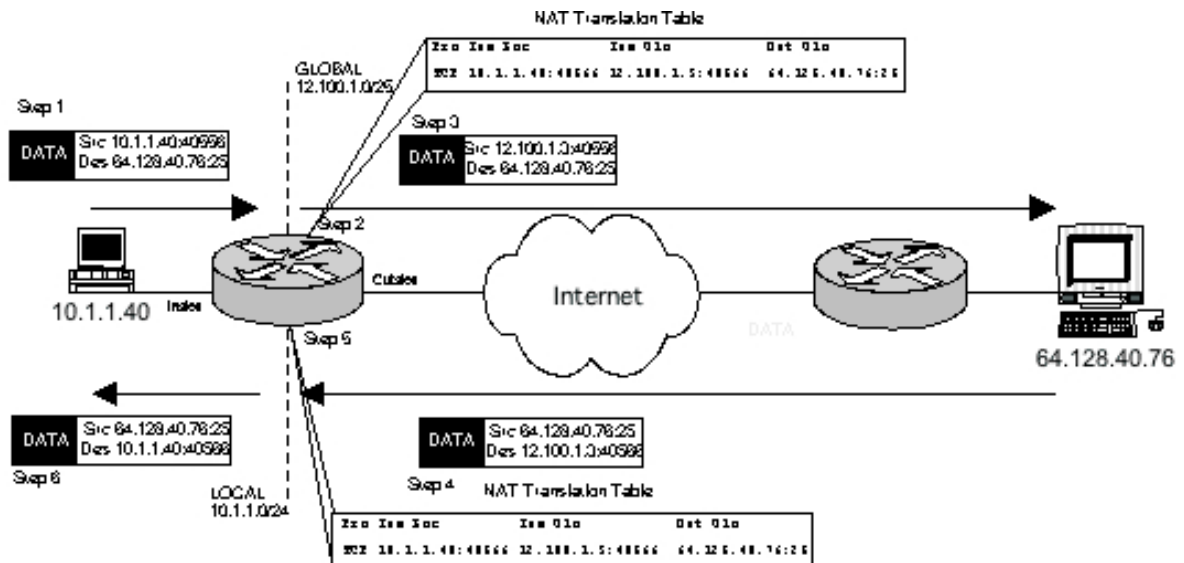
Michael J. Martin

to the Internet). Networks within the domain have direct reachability knowledge of one another through the use of Interior Gateway Protocols (IGP). The terms "public" and "private" are commonly used to describe the reachability status of a domain.

- A **public domain** or **outside network** is "reachable" from hosts outside of the direct administrative control of the domain and whose origin is not directly known (i.e., the Internet).
- A **private domain** or **inside network** is "reachable" only through an administrative gateway (AG). The AG is a device that monitors the communication session between the public and private host. AGs exist in many forms; firewalls, application proxy servers, or transparent routers are the most common AGs.
- **Address realm:** A contiguous IP address block (i.e. subnet) of an IP prefix within the network domain. Hosts within the realm are assigned specific addresses and have direct reachability to the local and remote networks within the domain.
- **Transparent routing:** The forwarding (i.e., translation) of IP packets between different address realms within a network domain. This is the technical function of NAT. All translations are passively performed without the knowledge of the communicating hosts, hence the name "transparent."
- A **local address** is a host address from an address realm within the network domain that is not directly reachable without transparent routing. Local address realms are not typically assigned by a regional routing registry (i.e., ARIN, RIPE, APIC), so the realm is not routable over the Internet. IANA has allocated the RFC 1918 address blocks for administrators to assign local addressing. While not advisable, it is possible for an administrator to use any IP prefix to assign local addresses.
- A **global address** is a host address from an address realm within the network domain that is reachable without transparent routing. Global address realms are assigned to network domains by an ISP or regional routing registry. Global addresses are "routable" over the Internet.
- **Session:** TCP is full-duplex transport protocol. Both the local and remote hosts are exchanging data over a virtual circuit, which needs to be set up and torn down. In contrast, UDP and ICMP are connectionless and send data in autonomous messages known as *datagrams*. The unidirectional exchanges between hosts are known as flows. Flows are a common way monitoring network activity, and flow has four essential parts: *src\_addr:src\_port:dst\_addr:dst\_port*. Data between local-to-remote and remote-to-local hosts is exchanged as a distinct flow, because NAT needs to perform both forward and reverse translations to facilitate data exchange. The NAT device tracks both unidirectional flows as an aggregate called a session. In addition, to the local and remote address and port information NAT also tracks the protocol (TCP, UDP, ICMP) as well. It uses a data set that looks something like this: *protocol:src\_addr:src\_port:dst\_addr:dst\_port*.

# Configuring IOS NAT

Michael J. Martin



Now that we've got the jargon down, let's run through the NAT process. A local /inside host initiates a connection with a global/remote host. This requires address translation, which can be broken down into a six-step process:

1. The local host with the transparent router (TR) configured as its default gateway (DG) performs a routing table lookup, ARP table lookup and forwards the packet to the DG.
2. The TR/DG accepts the packet on its inside interface. It performs a routing table lookup, determines the destination is outside of its domain and must forward the packet to its DG. The TR then checks its translation table (TT) to see if an outside global address has been associated with the inside local host. If an address is not found, it allocates one (which will remain allocated for an administratively determined period of time).
3. The TR then rewrites the packet's IP header, inserting the local address for the global address, and recalculates the IP and TCP header checksums to account for the new source address. Additionally, if the application inserts the packet's origin address within the data portion of the packet, these addresses must also be altered. This requirement makes not all applications compatible with NAT.
4. The remote host receives the packet and responds. When an inbound packet is received on the TR's global interface, the gateway checks the translation table and determines if the packet has a valid local destination. The "life" of a translation is determined administratively. TCP sessions, which have determinable start (SYN) and end points (SYN-RST, SYN-FIN), can be tracked and global address reallocation can be performed. The RFC recommends a minimum of 2 minutes. The maximum segment lifetime of 4 minutes should be used to avoid early session terminations in the case of network latency or retransmissions. UDP sessions, which are stateless, are not easy to track. Since UDP represents a problem from a security standpoint, a reasonable timeout should be considered (between 1 to 2 minutes).

# Configuring IOS NAT

Michael J. Martin

5. If a valid session match is found, the TR rewrites the header (and data) destination values and recalculates the needed checksums.
6. The router performs the needed routing and ARP lookups and forwards that packet to the local host.

## NAT flavors

Translation requirements differ, depending on network type and operational requirements. As a result, there are different flavors of NAT.

- **Bi-directional NAT**(also known as simple static NAT): Local-to-global address allocation is managed statically. Local-to-global mapping is created for each local host. Under the static model, reachability should be equivalent for local and global hosts, assuming the dynamic entries are knowable globally through a name resolution service such as DNS.
- **Traditional NAT**, dynamic NAT, outbound NAT, and simple NAT are just a few names for unidirectional NAT which allows non-reciprocal access for local hosts to access global hosts. Local-to-global address allocation can be either dynamic, using a pool of global addresses, or via static local-to-global mappings. Accessibility to the local hosts is limited to only locally originated sessions.
- **Twice NAT** or overlapping NAT is used when an administrator has used an illegal local address realm that has been legitimately allocated to an Internet entity, resulting in accessibility conflicts between the local realms and the global realm with the equivalent addressing. Twice NAT introduces a new type of local mapping known as **outside local** (an IOS construction for local-to-global translation). An external host has the same address of an inside local host, so any attempt to reach the external host using the conflicting address will fail. In this scenario, local hosts make requests to remote hosts with conflicting addresses using locally routable addresses as the destination address of the external hosts. When the router receives a packet with an outside local address, it translates it and forwards the request properly. For example, a Web site is globally addressed with a local address. A translation mapping using a locally routable address to the global address (in conflict) is made. Local DNS is configured to resolve the remote hostname to the global local address. When a connection is made, the router first translates the inside-local-to-outside-local mapping, then processes the inside-local-to-outside-global request.
- **Port Address Translation** (PAT) or Network Address Port Translation (NAPT) and overloaded NAT are NAT implementations that translate many host requests using a small outside global address pool or a single outside global address. Translations are multiplexed using inside local address and local transport port to outside global address and transport port.

At this point, you should know enough to properly interpret the IOS's configuration terminology. So let's move on to review some of the operational facets of the IOS NAT implementation and how to configure it.

# Configuring IOS NAT

Michael J. Martin

## Configuring IOS NAT

Cisco introduced RFC 1631-based NAT in IOS 11.2 in the "IP Plus" image tree. As of version 12.x, NAT is available in the "base" image tree and on all major hardware platforms. A small and home office implementation known as "Easy IP," which provides PAT only and provides DHCP server and interface configuration support, has been available since 11.3. NAT operates using "fast-switching" forwarding mode. NAT is processor and memory intensive. Each NAT table entry requires 160 bytes of DRAM, so the impact and scalability of a NAT implementation depends somewhat on the router platform and the amount of DRAM installed. Forwarding throughput varies depending on hardware platform. Because of NAT's operational nature, not all transport protocols and applications are supported. IP multicast, IPsec AH, NetShow and Interior Gateway Routing protocols are not supported. FTP, telnet, HTTP, SSH, RTTP, ICMP and DNS (UDP) series are just some of the supported applications. Application support varies depending on IOS, so check your version's release notes for a complete list.

IOS NAT configuration uses the exec and configuration command root `<ip nat>`. Basically, NAT configuration is two-step process: (1) definition of the inside and outside interfaces and (2) definition of inside local and outside global address mappings.

Let's start with definition of inside and outside network interfaces. IOS supports multiple inside interfaces and a single outside interface. On multi-interface routers, not all interfaces need to be defined. Undefined interfaces are considered "globally" accessible. Data exchanged between inside local hosts and hosts on undefined interfaces will be "translated." To configure an inside interface, the interface configuration sub-command `<ip nat inside>` is used. To configure an outside interface, the command `<ip nat outside>` is used. To disable NAT, use the `<no ip nat {inside|outside}>` variation of the command.

Once the interfaces have been defined, inside-local-to-outside-global address mappings need to be defined along with the NAT operational policy. The IOS supports three mapping methods: static local-to-global, dynamic pool allocation, and inside-local-to-outside-global interface. With the last option, the router's outside interface address is the global outside source address. Here are the steps to configure the available mapping and "inside-to-outside" NAT policy options:

1. To configure a static local inside to global outside mapping use the configuration command `<ip nat inside source static {out_glo_addr} {in_loc_addr}>`.
2. Configuring a dynamic pool is a three-step process: 1. Create a standard IP ACL to match the traffic to be translated `<access-list {1-99} {permit | deny} {host src_addr | any | src_addr wildcard mask}>`. 2. Create the outside global address pool using the configuration command `<ip local pool {pool name} {first addr in range} {last addr in range}>`. 3. Define the NAT policy to use dynamic-local-inside-to-global-outside mapping using the configuration command `< ip nat inside source list {acl #} pool {pool name} {overload}>`. When using a dynamic pool, once all of the outside global addresses have been allocated, no additional inside local hosts can establish connections with remote hosts. Including the `<overload>` enables source port translation. Local-to-global port uniqueness is maintained unless an existing connection using the same port is active. When a new port needs to be allocated, it is picked from the port same range according to the BSD

# Configuring IOS NAT

Michael J. Martin

allocation: Range #1 (1 ->511), range #2 (512->1023), range #3 (1024->4999), or range #4 (5000 -> 65535).

3. The inside-local-to-outside-global interface option was implemented on the IOS to support one-to-many PAT implementations. Inside local mappings can be configured statically to enable TCP and UDP port forwarding. Port forwarding allows an inside local host to receive inbound connections for specific services, such as HTTP and SMTP traffic, using the router's outside interface. To configure port forwarding, use the configuration command `<ip nat inside source static {tcp | udp} inside_loc_addr {port#}{interface x:x} {port#}>`. If you plan on forwarding telnet or SSH to access internal hosts, you cannot forward ports 22 and 23 because they are used by the router for VTY access. So you will have to use high-number ports as the outside global destination ports and map them to the known service port of the local inside host. For example, to set up SSH access to an inside host using port 4001, the configuration command is `<ip nat static tcp ins_loc_add 22 interface Ethernet 0/0 4001>`. Configuring outbound PAT follows steps similar to dynamic pool. First, create an inside local traffic qualifier map using a standard IP ACL with `<access-list {1-99} {permit | deny} {host src_addr | any | src_addr wildcard mask}>`. Then define the NAT policy using the configuration command `<ip nat inside source list {acl #} interface {L2 interface type x:x} overload>`.

Once NAT has been configured, there is little in terms of "hands-on" maintenance. That said, things can go wrong from time to time. To see a translation table, use the IOS exec command `<show ip nat translations>`. To see translation statistics (hits, misses, and expired translations) the exec command `<show ip nat statistics>` is available. To troubleshoot problems, a number of debug commands are available using the exec command root `<debug ip nat {option}>`. And if you are planning to make changes to the NAT policy once it is in place you will need to flush the translation table. This is done using an exec level `clear` command. To delete a NAT translation table entry, use the command `<clear ip nat translation {* | forced inside | outside}>`. The "\*" value clears all non-active dynamic translations, while the `forced` value clears everything. The `inside` value clears global static translations; `outside` clears local translations.

Now that we have reviewed the configuration commands, let's look at some excerpts of standard NAT configurations:

## SOHO Port Address Translation

For cable and ADSL users who do not get a static IP address assignment from their ISP, outbound PAT is the best option:

```
!  
version 12.2  
!  
hostname DCHP-Client-ID  
!  
!  
interface Ethernet0  
ip address dhcp  
ip access-group 100 in
```

# Configuring IOS NAT

Michael J. Martin

```
ip nat outside
half-duplex
!
interface FastEthernet0
ip address 192.168.1.1 255.255.255.0
ip nat inside
speed 100
duplex half
!
ip nat inside source list 2 interface Ethernet0 overload
!
!
access-list 2 permit 192.168.1.0 0.0.0.255
access-list 100 deny tcp any any eq 22
access-list 100 deny tcp any any eq telnet
access-list 100 deny tcp any any eq 80
access-list 100 permit icmp any any echo
access-list 100 permit icmp any any echo-reply
access-list 100 permit ip any any
!
!
```

Ethernet 0 is the ISP-side interface configured for DHCP address assignment. Most ISPs use the DHCP client ID as the authentication for getting an IP address on the network. In the example the <hostname> is set as the DHCP client ID, but it is also possible to set the DHCP client ID as part of the interface DHCP statement using <ip address dhcp client-id {value}>. There are two ACLs used in the example. Access list 2 defines the source addresses to be translated; it is always a good idea to be specific in this definition instead of using an "any" matching value. The other list is a basic inbound SACL filter. Your hosts may be hidden but your router is not. It is a good idea to block telnet, SSH and HTTP and only permit ICMP echo and echo-reply (if you really need them). Mistakes happen and most ISPs do not force the recycling of IP addresses. You could have the same IP address for a long time, so if someone is watching you could become a target.

## Traditional outbound NAT

This example is functionally similar to the PAT example, with the difference that inside hosts are assigned outside global addresses from a pool. The advantage with this approach is that traffic requests are coming from multiple IPs, making utilization harder to track back to specific inside hosts. Because this is an outbound-only solution, the same inbound filtering requirements on the "outside" interface should be followed.

```
!
version 12.2
!
hostname NAT-router
!
interface Ethernet0
```

# Configuring IOS NAT

Michael J. Martin

```
ip address 63.122.40.1 255.255.255.240
ip access-group 100 in
ip nat outside
half-duplex
!
interface FastEthernet0
ip address 192.168.100.1 255.255.255.0
ip nat inside
speed 100
duplex half
!
ip local pool nat 63.122.40.2 63.122.40.13
ip nat translation timeout 300
ip nat inside source list 2 pool nat overload
!
!
access-list 2 permit 192.168.1.0 0.0.0.255
access-list 100 deny tcp any any eq 22
access-list 100 deny tcp any any eq telnet
access-list 100 deny tcp any any eq 80
access-list 100 permit icmp any any echo
access-list 100 permit icmp any any echo-reply
access-list 100 permit ip any any
!
```

The use of the *<overload>* flag on the NAT policy is optional when using pools. But unless you have a global address pool that is one-to-one in ratio with your inside hosts, you may drop connections during high utilization. To minimize drops, it is a good idea to set the translation time (which is defined in seconds) out to around 5 minutes. If you are configuring NAT on an IOS version previous to 12.1, the pool command is different. In 12.1x, the command was changed to accommodate address pool grouping and overlapping. Previous to 12.1, only a single pool definition was permitted. The command syntax is *<ip nat pool {name} start-ip end-ip {netmask netmask | prefix-length prefix-length}>*. Notice the requirement for the netmask definition, which is used to verify which addresses within the pool can be assigned.

## Bi-directional NAT

This example shows a few of the static mapping options. A simple inside-to-outside bi-directional mapping (192.168.100.6 to 63.122.40.3) all traffic is permitted. The second static translation allows remote hosts to send SMTP mail and access POP mail. The third option provides round-robin load sharing to the site's Web server, which is hosted on two different systems. The fourth mapping example is a global inside mapping example that maps the public DNS server 204.22.6.100 to a local inside address (192.168.101.1). Notice that the actual "inside" address is not using same inside local IP prefix as the hosts and router. This is a routing trick, done because the hosts connected to the "local" subnets need to have the requests "forwarded" by the router in order for them to be processed.

# Configuring IOS NAT

Michael J. Martin

```
!  
version 12.2  
!  
hostname Router  
!  
!  
interface Ethernet0  
ip address 63.122.40.1 255.255.255.240  
ip nat outside  
half-duplex  
!  
interface FastEthernet0  
ip address 192.168.100.1 255.255.255.0  
ip nat inside  
speed auto  
!  
ip nat translation tcp-timeout 600  
ip nat translation udp-timeout 120  
!  
! One to One Mapping  
ip nat inside source static 192.168.100.6 63.122.40.3  
! Mail Server 25/110 Inbound Only Full Outbound  
ip nat inside source static tcp 192.168.100.10 25 63.122.40.4 25 extendable  
ip nat inside source static tcp 192.168.100.10 110 63.122.40.4 110  
extendable  
! Web Server Round Robin Load Balance 80 inbound only full outbound  
ip nat inside source static tcp 192.168.100.11 80 63.122.40.4 80 extendable  
ip nat inside source static tcp 192.168.100.12 80 63.122.40.4 80 extendable  
! Global Local mapping for Remote DNS quires  
ip nat inside source static 192.168.101.1 63.122.40.5  
ip nat outside source static 204.22.6.100 192.168.101.1  
!  
!  
!
```

## IPsec ESP tunneling NAT

This last example shows IPsec Encapsulation Security Payload (ESP) tunneling with outbound PAT. ESP tunneling facilitates inside local host connections with remote IPsec/ESP gateways. Tunneling NAT (TNAT) only works for ESP Authentication Header (AH). IPsec policies will not work with TNAT because the host's origin address is encrypted and cannot be modified by NAT. Notice once again the also outside local translation for a remote DNS server, which minimizes the use of UDP translation services by inside local hosts.

```
!  
version 12.2  
!
```

# Configuring IOS NAT

Michael J. Martin

```
hostname ADSL-GW
!
!
interface Ethernet0
ip address 192.168.1.1 255.255.255.0
ip nat inside
half-duplex
!
interface FastEthernet0
ip address 12.100.2.253 255.255.255.252
ip nat outside
!
! Generic PAT translation policy
ip nat inside source list 2 interface FastEthernet0 overload
! TNAT statement for inside local host 192.168.1.3
ip nat inside source static esp 192.168.1.3 interface FastEthernet0
! Global Local mapping for Remote DNS quires
ip nat outside source static 216.140.16.254 192.168.2.100
!
access-list 2 permit 192.168.1.3
!
!
```

## Conclusion

Jargon aside, as you can see from the examples, IOS NAT is easy to implement. The hope is that your understanding of NAT and some of its implementation ticks and options have been made clearer. Tune in again next month, when I will take a look at the IOS's Dynamic Host Configuration Protocol (DHCP) client and server supported options, when to use them (risks and rewards) and how to configure them.