Networking Protocols

5.5.1

Networking Protocols

The internet was built on various standards. You should understand the standard network protocols so you can communicate and troubleshoot effectively.

Each protocol meets a need and uses standard port values. You should know when to use a particular protocol and know the standard port for connections. Many developers have been puzzled by a mismatched port value; therefore, checking these values can be a first line of attack when troubleshooting.

Telnet and Secure SHell (SSH)

Telnet and SSH are both used to connect to a remote computer and log in to that system using credentials. Telnet is less prevalent today because SSH uses encryption to protect data going over the network connection. Telnet should only be used in non-production environments.

SSH connections can use a public key for authentication, rather than sending a username and password over the network. This authentication method means that SSH is a good choice to connect to network devices, to cloud devices, and to containers.

By default, SSH uses port 22 and Telnet uses port 23. Telnet can use port 992 when creating a session over Transport Layer Security (TLS) or SSL.

HTTP and HTTPS

HTTP and its secure version, HTTPS, are both protocols recognized by web browsers and are used to connect to web sites. HTTPS uses TLS or SSL to make a secure connection. You can see the **http:** or **https:** in the address bar on your browser. Many browsers also recognize **ssh:** and **ftp:** protocols and allow you to connect to remote servers in that way as well.

NETCONF and RESTCONF

Later in this course, you will use NETCONF and RESTCONF to manage a Cisco router. NETCONF uses port 830. RESTCONF does not have a reserved port value. You may see various implementations of different values. Commonly the port value is in the 8000s.

To have multiple network operations, you want to make sure each protocol has a default port and use standards to try to avoid conflicts. TCP and UDP traffic requires a destination port be specified for each packet. The source port is automatically generated by the sending device. The following table shows some common, well-known port values for protocols used in this course. System port numbers are in the range 0 to 1023, though you may see others in use for different reasons.

Note: For a more complete list of ports, search the internet for TCP and UPD port numbers.

Port value	Protocol
22	SSH
23, 992	Telnet
53	DNS
80	HTTP
443	HTTPS (HTTP over TLS or SSL)
830	NETCONF
8008, 8080, 8888	RESTCONF

5.5.2

DHCP

As you have seen previously in this module, IP addresses are needed by all devices connected to a network in order for them to be able to communicate. Assigning these IP addresses manually and one at a time for each device on the network is cumbersome and time consuming. DHCP was designed to dynamically configure devices with IP addressing information. DHCP works within a client/server model, where designated DHCP servers allocate IP addresses and deliver configuration information to devices that are configured to dynamically request addressing information.

In addition to the IP address for the device itself, a DHCP server can also provide additional information, like the IP address of the DNS server, default router, and other configuration parameters. For example, Cisco wireless access points use option 43 in DHCP requests to obtain the IP address of the Wireless LAN Controller that they need to connect to for management purposes.

Some of the benefits of using DHCP instead of manual configurations are:

- **Reduced client configuration tasks and costs** – By not having to physically walk up to the device and manually configure the network settings, large cost savings are possible. This especially applies in the case of ISPs that can remotely and dynamically assign IP addresses to the cable or Digital Subscriber Line (DSL) modems of their clients without having to dispatch a person each time a network configuration change is necessary.
- **Centralized management** – A DHCP server typically maintains the configuration settings for several subnets. Therefore, an administrator only needs to configure and update a single, central server.

DHCP allocates IP addresses in three ways:

- **Automatic allocation** - The DHCP server assigns a permanent IP address to the client.
- **Dynamic allocation** - DHCP assigns an IP address to a client for a limited period of time (lease time).
- **Manual allocation** - The network administrator assigns an IP address to a client and DHCP is used to relay the address to the client.

DHCP defines a process by which the DHCP server knows the IP subnet in which the client resides, and it can assign an IP address from a pool of available addresses in that subnet. The rest of the network configuration parameters that a DHCP server supplies, like the default router IP address or the IP address of the DNS server, are usually the same for the whole subnet so the DHCP server can have these configurations per subnet rather than per host.

The specifications for the IPv4 DHCP protocol are described in *RFC 2131 - Dynamic Host Configuration Protocol* and *RFC 2132 - DHCP options and BOOTP Vendor Extensions*. DHCP for IPv6 was initially described in *RFC 3315 - Dynamic Host Configuration Protocol for IPv6 (DHCPv6)* in 2003, but this has been updated by several subsequent RFCs. *RFC 3633 - IPv6 Prefix Options for Dynamic Host Configuration Protocol (DHCP) version 6* added a DHCPv6 mechanism for prefix delegation and *RFC 3736 - Stateless Dynamic Host Configuration Protocol (DHCP) Service for IPv6* added SLAAC. The main difference between DHCP for IPv4 and DHCP for IPv6 is that DHCP for IPv6 does not include the default gateway address. The default gateway address can only be obtained automatically in IPv6 from the Router Advertisement message.

DHCP Relay

In cases in which the DHCP client and server are located in different subnets, a DHCP relay agent can be used. A relay agent is any host that forwards DHCP packets between clients and servers. Relay agent forwarding is different from the normal forwarding that an IP router performs, where IP packets are routed between networks transparently. Relay agents receive inbound DHCP messages and then generate new DHCP messages on another interface, as shown in the figure.

DHCP Relay

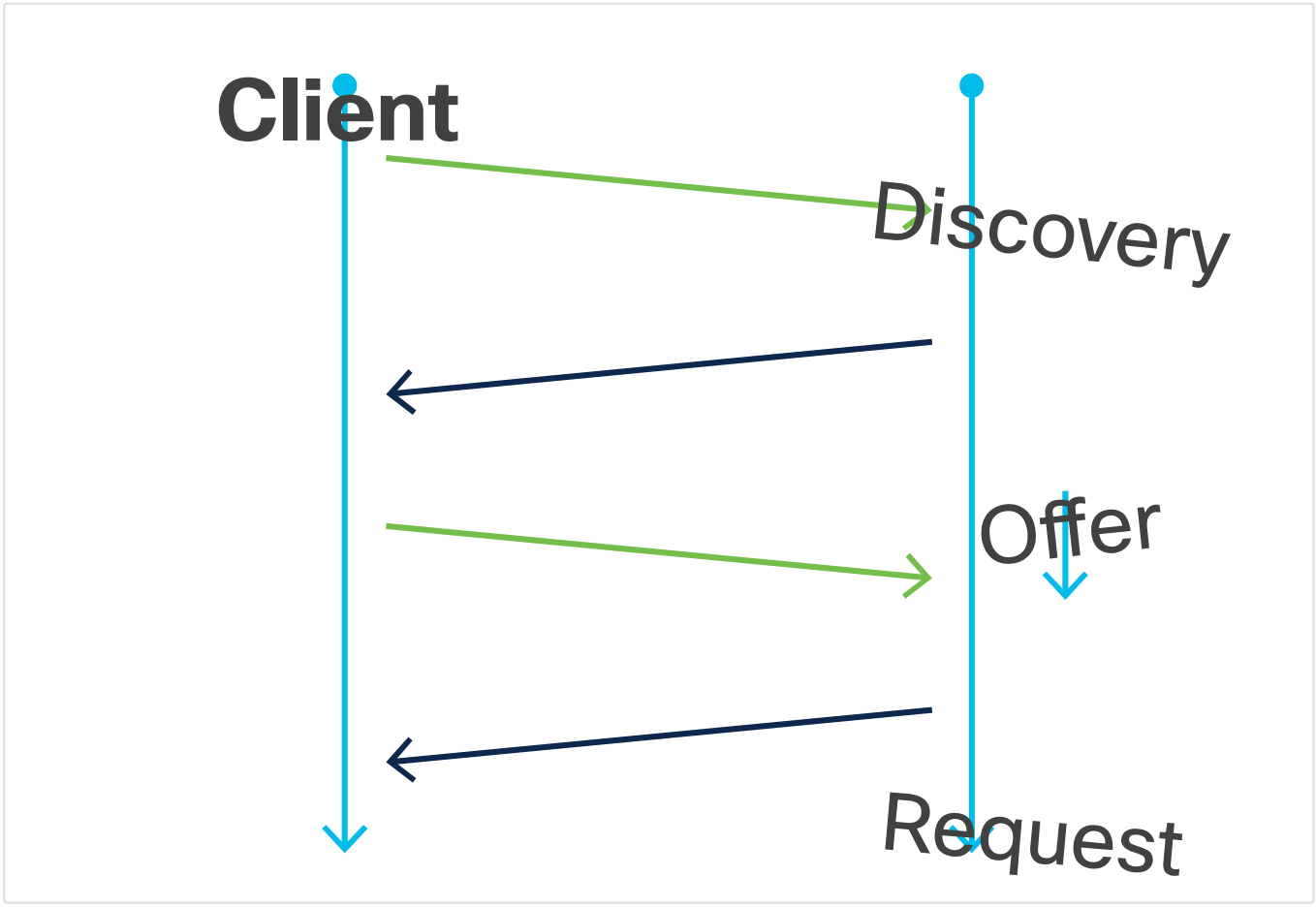
Clients use port 67 to send DHCP messages to DHCP servers. DHCP servers use port 68 to send DHCP messages to clients.

DHCP operations includes four messages between the client and the server:

- DHCPDISCOVER - Server discovery
- DHCPPOFFER - IP lease offer
- DHCPREQUEST - IP lease request
- DHCPACK - IP lease acknowledgment

The figure shows how these messages are sent between the client and server.

DHCP Operations



In the figure, the client broadcasts a DHCPDISCOVER message looking for a DHCP server. The server responds with a unicast DHCPOFFER. If there is more than one DHCP server on the local network, the client may receive multiple DHCPOFFER messages. Therefore, it must choose between them, and broadcast a DHCPREQUEST message that identifies the explicit server and lease offer that the client is accepting. The message is sent as a broadcast so that any other DHCP servers on the local network will know the client has requested configuration from another DHCP server.

A client may also choose to request an address that it had previously been allocated by the server. Assuming that the IPv4 address requested by the client, or offered by the server, is still available, the server sends a unicast DHCP acknowledgment (DHCPACK) message that acknowledges to the client that the lease has been finalized. If the offer is no longer valid, then the selected server responds with a DHCP negative acknowledgment (DHCPNAK) message. If a DHCPNAK message is returned, then the selection process must begin again with a new DHCPDISCOVER message. After the client has the lease, it must be renewed prior to the lease expiration through another DHCPREQUEST message.

The DHCP server ensures that all IP addresses are unique (the same IP address cannot be assigned to two different network devices simultaneously). Most ISPs use DHCP to allocate addresses to their customers.

DHCPv6 has a set of messages that is similar to those for DHCPv4. The DHCPv6 messages are SOLICIT, ADVERTISE, INFORMATION REQUEST, and REPLY.

5.5.3

DNS

In data networks, devices are labeled with numeric IP addresses to send and receive data over networks. Domain names were created to convert the numeric address into a simple, recognizable name.

On the internet, fully-qualified domain names (FQDNs), such as http://www.cisco.com, are much easier for people to remember than 198.133.219.25, which is the actual numeric address for this server. If Cisco decides to change the numeric address of www.cisco.com, it is transparent to the user because the domain name remains the same. The new address is simply linked to the existing domain name and connectivity is maintained.

Note: You will not be able to access www.cisco.com by simply entering that IP address 198.133.219.25 in your web browser.

The DNS protocol defines an automated service that matches domain names to IP addresses. It includes the format for queries, responses, and data. DNS uses a single format called a DNS message. This message format is used for all types of client queries and server responses, error messages, and the transfer of resource record information between servers.

DNS Message Format

The DNS server stores different types of resource records that are used to resolve names. These records contain the name, address, and type of record. Some of these record types are as follows:

- A – An end device IPv4 address
- NS – An authoritative name server
- AAAA – An end device IPv6 address (pronounced quad-A)
- MX – A mail exchange record

When a client makes a query to its configured DNS server, the DNS server first looks at its own records to resolve the name. If it is unable to resolve the name by using its stored records, it contacts other servers to resolve the name. After a match is found and returned to the original requesting server, the server temporarily stores the numbered address in the event that the same name is requested again.

The DNS client service on Windows PCs also stores previously resolved names in memory. The `ipconfig /displaydns` command displays all of the cached DNS entries.

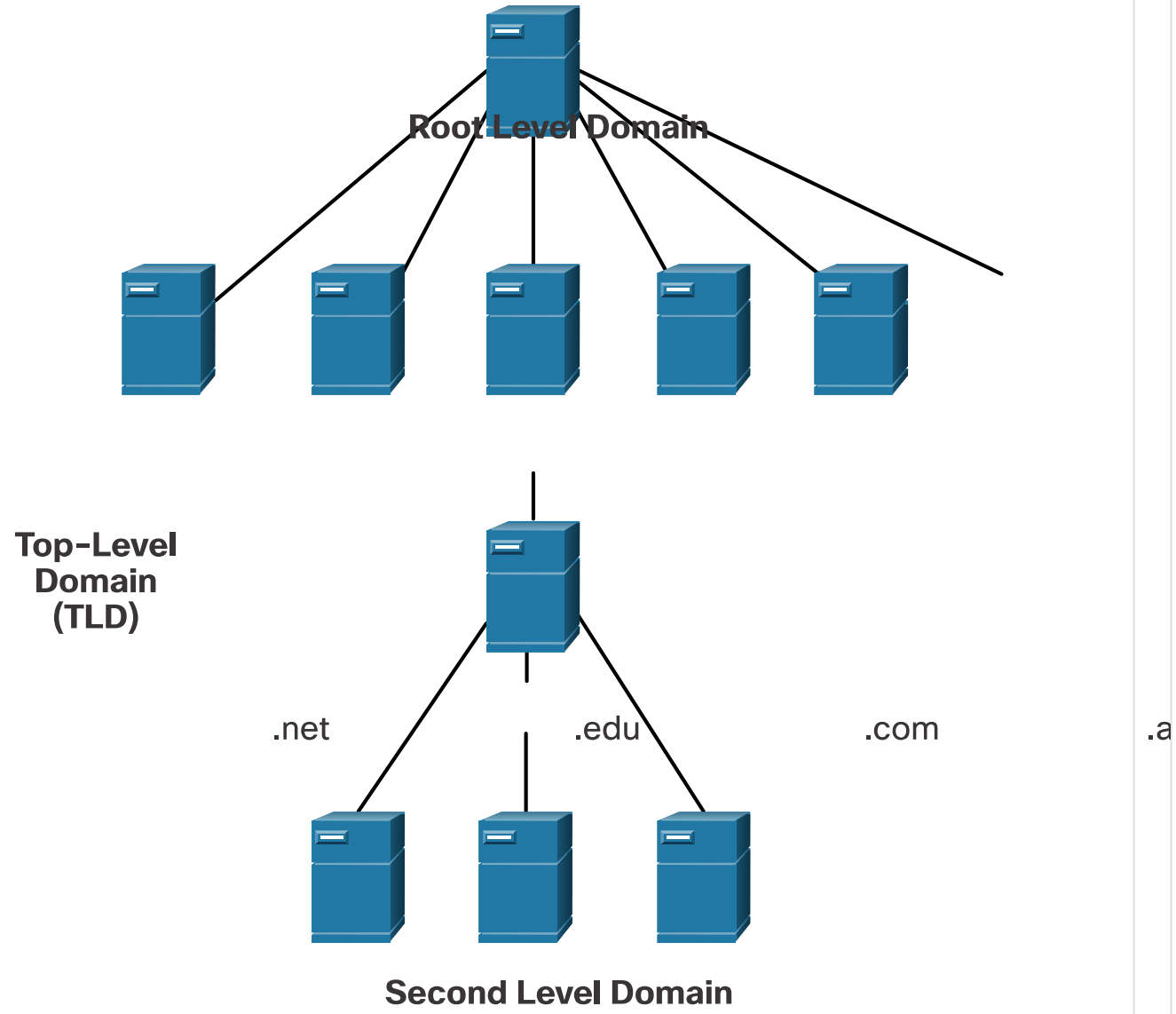
As shown in the table, DNS uses the same message format between servers. It consists of a question, answer, authority, and additional information for all types of client queries and server responses, error messages, and transfer of resource record information.

DNS Message Section	Description
Question	The question for the name server
Answer	Resource Records answering the question

DNS Message Section	Description
Authority	Resource Records pointing toward an authority
Additional	Resource Records holding additional information

DNS Hierarchy

DNS uses a hierarchical system based on domain names to create a database to provide name resolution, as shown in the figure.



The naming structure is broken down into small, manageable zones. Each DNS server maintains a specific database file and is only responsible for managing name-to-IP mappings for that small portion of the entire DNS structure. When a DNS server receives a request for a name translation that is not within its DNS zone, the DNS server forwards the request to another DNS server within the proper zone for translation. DNS is scalable because hostname resolution is spread across multiple servers.

The different top-level domains represent either the type of organization or the country of origin. Examples of top-level domains are the following:

- .com – a business or industry
- .org – a non-profit organization
- .au – Australia
- .co – Colombia

www.cisco.com ftp.cisco.com mail.ci

Note: For more examples, search the internet for a list of all the top-level domains.

5.5.4

SNMP

SNMP was developed to allow administrators to manage devices such as servers, workstations, routers, switches, and security appliances. It enables network administrators to monitor and manage network performance, find and solve network problems, and plan for network growth. SNMP is an application layer protocol that provides a message format for communication between managers and agents.

There are several versions of SNMP that have been developed through the years:

- SNMP Version 1 (SNMPv1)
- SNMP Version 2c (SNMPv2c)
- SNMP Version 3 (SNMPv3)

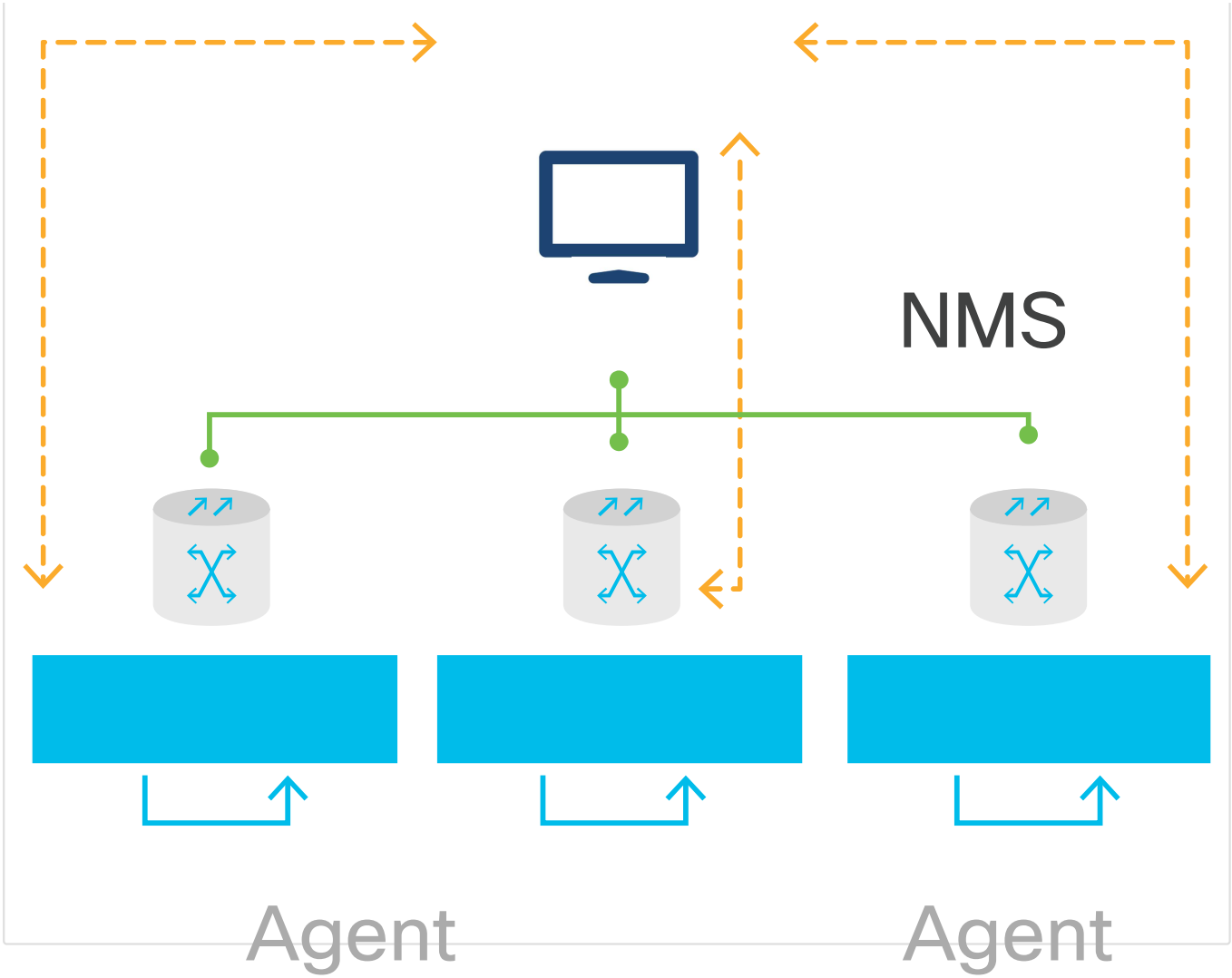
SNMP version 1 is rarely used anymore, but versions 2c and 3 are still extensively used. In comparison to previous versions, SNMPv2c includes additional protocol operations and 64-bit performance monitoring support. SNMPv3 focused primarily on improving the security of the protocol. SNMPv3 includes authentication, encryption, and message integrity.

The SNMP system consists of three elements:

- SNMP manager: network management system (NMS)
- SNMP agents (managed device)
- Management Information Base (MIB)

The figure shows the relationship(s) among SNMP manager, agents, and managed devices.

SNMP Components



To configure SNMP on a networking device, it is first necessary to define the relationship between the SNMP manager and the device (the agent).

The SNMP manager is part of a network management system (NMS). The SNMP manager runs SNMP management software. As shown in the figure, the SNMP manager can collect information from an SNMP agent by using the “get” action. It can also change configurations on an agent by using the “set” action. In addition, SNMP agents can forward information directly to the SNMP manager by using “traps”.

An SNMP agent running on a device collects and stores information about the device and its operation. This information is stored locally by the agent in the MIB. The SNMP manager then uses the SNMP agent to access information within the MIB and make changes to the device configuration.

There are two primary SNMP manager requests, **get** and **set**. A **get** request is used by the SNMP manager to query the device for data. A **set** request is used by the SNMP manager to change configuration variables in the agent device. A **set** request can also initiate actions within a device. For example, a **set** can cause a router to reboot, send a configuration file, or receive a configuration file.

SNMP Polling

The NMS can be configured to periodically have the SNMP managers poll the SNMP agents that are residing on managed devices using the **get** request. The SNMP manager queries the device for data. Using this process, a network management application can collect information to monitor traffic loads and to verify the device configurations of managed devices. The information can be displayed via a GUI on the NMS. Averages, minimums, or maximums can be calculated. The data can be graphed, or thresholds can be established to trigger a notification process when the thresholds are exceeded. For example, an NMS can monitor CPU utilization of a Cisco router. The SNMP manager samples the value periodically and presents this information in a graph for the network administrator to use in creating a baseline, creating a report, or viewing real time information.

SNMP Traps

Periodic SNMP polling does have disadvantages. First, there is a delay between the time that an event occurs and the time that it is noticed (via polling) by the NMS. Second, there is a trade-off between polling frequency and bandwidth usage.

To mitigate these disadvantages, it is possible for SNMP agents to generate and send traps to inform the NMS immediately of certain events. Traps are unsolicited messages alerting the SNMP manager to a condition or event on the network. Examples of trap conditions include, but are not limited to, improper user authentication, restarts, link status (up or down), MAC address tracking, closing of a TCP connection, loss of connection to a neighbor, or other significant events. Trap notifications reduce network and agent resources by eliminating the need for some of SNMP polling requests.

SNMP Community Strings

For SNMP to operate, the NMS must have access to the MIB. To ensure that access requests are valid, some form of authentication must be in place.

SNMPv1 and SNMPv2c use community strings that control access to the MIB. Community strings are plaintext passwords. SNMP community strings authenticate access to MIB objects.

There are two types of community strings:

- **Read-only (ro)** – This type provides access to the MIB variables, but does not allow these variables to be changed. Because security is minimal in version 2c, many organizations use SNMPv2c in read-only mode.
- **Read-write (rw)** – This type provides read and write access to all objects in the MIB.

To get or set MIB variables, the user must specify the appropriate community string for read or write access.

Management Information Base (MIB)

The agent captures data from MIBs, which are data structures that describe SNMP network elements as a list of data objects. Think of the MIB as a "map" of all the components of a device that are being managed by SNMP. To monitor devices, the SNMP manager must compile the MIB file for each equipment type in the network. Given an appropriate MIB, the agent and SNMP manager can use a relatively small number of commands to exchange a wide range of information with one another.

The MIB is organized in a tree-like structure with unique variables represented as terminal leaves. An Object IDentifier (OID) is a long numeric tag. It is used to distinguish each variable uniquely in the MIB and in the SNMP messages. Variables that measure things such as CPU temperature, inbound packets on an interface, fan speed, and other metrics, all have associated OID values. The MIB associates each OID with a human-readable label and other parameters, serving as a dictionary or codebook. To obtain a metric (such as the state of an alarm, the host name, or the device uptime), the SNMP manager puts together a **get** packet that includes the OID for each object of interest. The SNMP agent on the device receives the request and looks up each OID in its MIB. If the OID is found, a response packet is assembled and sent with the current value of the object included. If the OID is not found, an error response is sent that identifies the unmanaged object.

SNMP traps are used to generate alarms and events that are happening on the device. Traps contain:

- OIDs that identify each event and match it with the entity that generated the event
- Severity of the alarm (critical, major, minor, informational or event)
- A date and time stamp

SNMP Communities

SNMP community names are used to group SNMP trap destinations. When community names are assigned to SNMP traps, the request from the SNMP manager is considered valid if the community name matches one configured on the managed device. If so, all agent-managed MIB variables are made accessible.

If the community name does not match, however, SNMP drops the request. New devices are often preconfigured to have SNMP enabled, and provided with basic communities named public for read-only access and private for read/write access to the system. From a security perspective, is very important to either rename these communities, remove them completely and disable SNMP if not used or apply an access-list to the community, limiting the access to only the IP address or hostname of the SNMP manager station.

SNMP Messages

SNMP uses the following messages to communicate between the manager and the agent:

- Get
- GetNext
- GetResponse
- Set
- Trap

The Get and GetNext messages are used when the manager requests information for a specific variable. When the agent receives a Get or GetNext message it will issue a GetResponse message back

to the manager. The response message will contain either the information requested or an error message indicating why the request cannot be processed.

A Set message is used by the manager to request that a change should be made to the value of a specific variable. Similarly, to the Get and GetNext requests, the agent will respond with a GetResponse message indicating either that the change has been successfully done or an error message indicating why the requested change cannot be implemented.

The Trap message is used by the agent to inform the manager when important events take place. An SNMP Trap is a change of state message. This means it could be one of the following:

- an alarm
- a clear
- a status message

Several Requests for Comments (RFCs) have been published throughout the years concerning SNMP. Some notable ones are: *RFC 1155 - Structure and Identification of Management Information for the TCP/IP-based Internets*, *RFC 1213 - Management Information Base for Network Management of TCP/IP-based Internets: MIB-II* and *RFC 2578 - Structure of Management Information Version 2 (SNMP)*.

5.5.5

NTP

Accurate time and making sure all devices in the network have a uniform and correct view of time has always been a critical component to ensuring a smooth operation of the infrastructure. IT infrastructure, including the network, compute, and storage, has become critical for the success of nearly all businesses today. Every second of downtime or unavailability of services over the network can be extremely expensive. In cases where these issues extend over hours or days it can mean bankruptcy and going out of business. Service Level Agreements (SLAs) are contracts between parties that consume infrastructure services and parties that provide these services. Both parties depend on accurate and consistent timing from a networking perspective. Time is fundamental to measuring SLAs and enforcing contracts.

The system clock on each device is the heart of the time service. The system clock runs from the second the operating system starts and keeps track of the date and time. The system clock can be set to update from different sources and can be used to distribute time to other systems through various mechanisms. Most network devices contain a battery-powered clock to initialize the system clock. The battery-powered clock tracks date and time across restarts and power outages. The system clock keeps track of time based on Universal Time Code (UTC), equivalent to Greenwich Mean Time (GMT). Information about local timezone and regional daylight savings time can be configured to enable display of local time and date wherever the server is located.

NTP Overview

Network Time Protocol (NTP) enables a device to update its clock from a trusted network time source, compensating for local clock drift. A device receiving authoritative time can be configured to serve time to other machines, enabling groups of devices to be closely synchronized.

NTP uses UDP port 123 as source and destination. RFC 5905 contains the definition of NTP Version 4, which is the latest version of the protocol. NTP is used to distribute and synchronize time among distributed time servers and clients. A group of devices on a network that are configured to distribute NTP and the devices that are updating their local time from these time servers form a synchronization subnet. Multiple NTP time masters (primary servers) can exist in the same synchronization subnet at the same time. NTP does not specify any election mechanism between multiple NTP servers in the same synchronization subnet. All available NTP servers can be used for time synchronization at the same time.

An authoritative time source is usually a radio clock, or an atomic clock attached to a time server. Authoritative server in NTP lingo just means a very accurate time source. It is the role of NTP to distribute the time across the network. NTP clients poll groups of time servers at intervals managed dynamically to reflect changing network conditions (primarily latency) and the judged accuracy of each time server consulted (determined by comparison with local clock time). Only one NTP transaction per minute is needed to synchronize the time between two machines.

NTP uses the concept of strata (layers) to describe how far away a host is from an authoritative time source. The most authoritative sources are in stratum 1. These are generally servers connected directly to a very accurate time source, like a rubidium atomic clock. A stratum 2 time server receives time from a stratum 1 server, and so on. When a device is configured to communicate with multiple NTP servers, it will automatically pick the lowest stratum number device as its time source. This strategy builds a self-organizing tree of NTP speakers. NTP performs well over packet-switched networks like the internet, because it makes correct estimates of the following three variables in the relationship between a client and a time server:

- Network delay between the server and the client.
- Dispersion of time data exchanges. Dispersion represents a measure of the maximum clock error between the server and the client.
- Clock offset, which is the correction applied to a client's clock to synchronize it to the current time.

It is not uncommon to see NTP clock synchronization at the 10 millisecond level over long distance WANs with devices as far apart as 2000 miles, and at the 1 millisecond level for LANs.

NTP avoids synchronizing with upstream servers whose time is not accurate. It does this in two ways:

- NTP never synchronizes with a NTP server that is not itself synchronized.
- NTP compares time reported by several NTP servers, and will not synchronize to a server whose time is an outlier, even if its stratum is lower than the other servers' stratum.

Communication between devices running NTP, also known as associations, are usually statically configured. Each device is given the IP address or hostname of all NTP servers it should associate with, and connects with them directly to solicit time updates. In a LAN environment, NTP can be configured to use IP broadcast messages instead. Configuration complexity is reduced in this case because each device can be configured to send or receive broadcast messages. The downside with this situation is that the accuracy of timekeeping is a bit reduced because the information flow is only one-way.

The time kept on a device is a critical resource. It is strongly recommended to use the security features that come with NTP to avoid the accidental or malicious configuration of incorrect time. The two security features usually used are:

- An access list-based restriction scheme in which NTP traffic is allowed in the network only from specific sources.
- An encrypted authentication mechanism in which both the clients and the servers authenticate each other securely.

Clients usually synchronize with the lowest stratum server they can access. But NTP incorporates safeguards as well: it prefers to have access to at least three lower-stratum time sources (giving it a quorum), because this helps it determine if any single source is incorrect. When all servers are well synchronized, NTP chooses the best server based on a range of variables: lowest stratum, network distance (latency), and precision claimed. This suggests that while one should aim to provide each client with three or more sources of lower stratum time, it is not necessary that all these sources be of highest quality. For example, good backup service can be provided by a same-stratum peer that receives time from different lower stratum sources.

In order to determine if a server is reliable, the client applies many sanity checks:

- Timeouts to prevent trap transmissions if the monitoring program does not renew this information after a lengthy interval.
- Checks on authentication, range bounds, and to avoid use of very old data.
- Checks warn that the server's oscillator (local clock tick-source) has gone too long without update from a reference source.
- Recent additions to avoid instabilities when a reference source changes rapidly due to severe network congestion.

If any one of these checks fail, the device declares the source insane.

NTP Association Modes

NTP servers can associate in several modes, including:

- Client/Server
- Symmetric Active/Passive
- Broadcast

Client/Server Mode

Client/server mode is most common. In this mode, a client or dependent server can synch with a group member, but not the reverse, protecting against protocol attacks or malfunctions.

Client-to-server requests are made via asynchronous remote procedure calls, where the client sends a request and expects a reply at some future time (unspecified). This is sometimes described as "polling". On the client side, client/server mode can be turned on with a single command or config-file change, followed by a restart of the NTP service on the host. No additional configuration is required on the NTP server.

In this mode, a client requests time from one or more servers and processes replies as received. The server changes addresses and ports, overwrites certain message fields, recalculates the checksum,

and returns the message immediately. Information included in the NTP message lets the client determine the skew between server and local time, enabling clock adjustment. The message also includes information to calculate the expected timekeeping accuracy and reliability, as well as help the client select the best server.

Servers that provide time to many clients normally operate as a three-server cluster, each deriving time from three or more stratum 1 or 2 servers as well as all other members of the group. This protects against situations where one or more servers fail or become inaccurate. NTP algorithms are designed to identify and ignore wrongly-functioning time sources and are even resilient against attacks where NTP servers are subverted deliberately and made to send incorrect time to clients. As backup, local hosts can be equipped with external clocks and made to serve time temporarily, in case normal time sources, or communications paths used to reach them, are disrupted.

Symmetric Active/Passive Mode

In this mode, a group of low stratum peers work as backups for one another. Each peer derives time from one or more primary reference sources or from reliable secondary servers. As an example, a reference source could be a radio clock, receiving a corrected time signal from an atomic clock. Should a peer lose all reference sources or stop working, the other peers automatically reconfigure to support one another. This is called a 'push-pull' operation in some contexts: peers either pull or push time, depending on self-configuration.

Symmetric/active mode is usually configured by declaring a peer in the configuration file, telling the peer that one wishes to obtain time from it, and provide time back if necessary. This mode works well in configurations where redundant time servers are interconnected via diverse network paths, which is a good description of how most stratums 1 and 2 servers on the internet are set up today.

Symmetric modes are most often used to interconnect two or more servers that work as a mutually redundant group. Group members arrange their synch paths to minimize network jitter and propagation delay.

Configuring a peer in symmetric/active mode is done with the peer command, and then providing the DNS name or address of the other peer. The other peer may also be configured in symmetric active mode in this way.

If not, a symmetric passive association is activated automatically when a symmetric active message is received. Because intruders can impersonate a peer and inject false time values, symmetric mode should always be authenticated.

Broadcast and/or Multicast Mode

When only modest requirements for accuracy exist, clients can use NTP broadcast and/or multicast modes, where many clients are configured the same way, and one broadcast server (on the same subnet) provides time for them all. Broadcast messages are not propagated by routers, meaning that this mode cannot be used beyond a single subnet.

Configuring a broadcast server is done using the broadcast command, and then providing a local subnet address. The broadcast client command lets the broadcast client respond to broadcast messages received on any interface. This mode should always be authenticated, because an intruder can impersonate a broadcast server and propagate false time values.

5.5.6

NAT

Although implementation of IPv6 and its 40 undecillion addresses is proceeding, IPv4 is still widely used. IPv4 can only accommodate a maximum of slightly over 4 billion unique addresses (2 to the 32nd power). This creates problems. Given the necessarily-limited range of public or external IPv4 addresses an organization (or subnetwork) can control, how can many more devices use these addresses to communicate outside?

Network Address Translation (NAT) helps with the problem of IPv4 address depletion. NAT works by mapping many private internal IPv4 addresses to a range of public addresses or to one single address (as is done in most home networks). NAT identifies traffic to and from a specific device, translating between external/public and internal/private IPv4 addresses.

NAT also hides clients on the internal network behind a range of public addresses, providing a "sense of security" against these devices being directly attacked from outside. As mentioned previously, the IETF does not consider private IPv4 addresses or NAT as effective security measures.

NAT is supported on a large number of routers from different vendors for IPv4 address simplification and conservation. In addition, with NAT you can select which internal hosts are available for NAT and hence external access.

NAT can be configured on hosts and routers requiring it, without requiring any changes to hosts or routers that do not need NAT. This is an important advantage.

Purpose of NAT

By mapping between external and internal IPv4 addresses, NAT allows an organization with non-globally-routable IPv4 addresses to connect to the internet by translating addresses into a globally-routable IPv4 address. Non-globally-routable addresses, or private addresses are defined by RFC 1918 (10.0.0.0/8, 172.16.0.0/12, 192.168.0.0/16). These addresses are private and cannot be routed on the internet. With the exception of a few other types of IPv4 addresses, other IPv4 addresses are globally-routable on the internet. They are known as public IPv4 addresses.

NAT also enables an organization to easily change service providers or voluntarily renumber network resources without affecting their public IPv4 address space. NAT is an IETF standard and is described in *RFC 1631 - The IP Network Address Translator (NAT)*.

NAT is typically configured at the point of connection between the internal network and the outside network or the internet. For each packet exiting the domain, NAT translates the source address into a globally unique address, and vice-versa. Networks with more than one point of entrance/exit require multiple NATs, sharing the same translation table. If NAT runs out of addresses from the pool, it drops the packet and sends an ICMP host unreachable message to the destination.

Used in the context of NAT, the term "inside" usually means networks controlled by an organization, with addresses in one local address space. "Outside" refers to networks to which the network connects, and which may not be under the organization's control. Hosts in outside networks may also be subject to translation, with their own local and global IPv4 addresses.

Types of NAT

NAT typically runs on a router. Before packets are forwarded between networks, NAT translates the private (inside local) addresses within the internal network into public (inside global) addresses. This functionality gives the option to configure NAT so that it advertises only a single address to the outside world, for the entire internal network. By so doing, NAT effectively hides the internal network from the world.

Types of NAT include:

- **Static address translation (static NAT)** – This is one-to-one mapping between global and local IPv4 addresses.
- **Dynamic address translation (dynamic NAT)** – This maps registered IPv4 addresses from a pool to registered IP addresses.
- **Overloading (also called Port Address Translation or PAT)** – This maps many unregistered IPv4 addresses to a single registered address (many to one) on different ports. Through overloading, thousands of users can be connected to the internet by using only one real global IP address.

IPv6 was developed with the intention of making NAT unnecessary. However, IPv6 does include its own IPv6 private address space called unique local addresses (ULAs). IPv6 ULAs are similar to RFC 1918 private addresses in IPv4 but have a different purpose. ULAs are meant for only local communications within a site. ULAs are not meant to provide additional IPv6 address space, nor to provide a level of security.

IPv6 does provide for protocol translation between IPv4 and IPv6. This is known as NAT64. NAT for IPv6 is used in a much different context than NAT for IPv4. The varieties of NAT for IPv6 are used to transparently provide access between IPv6-only and IPv4-only networks. It is not used as a form of private IPv6 to global IPv6 translation.

Four NAT Addresses

NAT includes four types of addresses:

- Inside local address
- Inside global address
- Outside local address
- Outside global address

When determining which type of address is used, it is important to remember that NAT terminology is always applied from the perspective of the device with the translated address:

- **Inside address** – This is the address of the device which is being translated by NAT.
- **Outside address** – This is the address of the destination device.

NAT also uses the concept of local or global with respect to addresses:

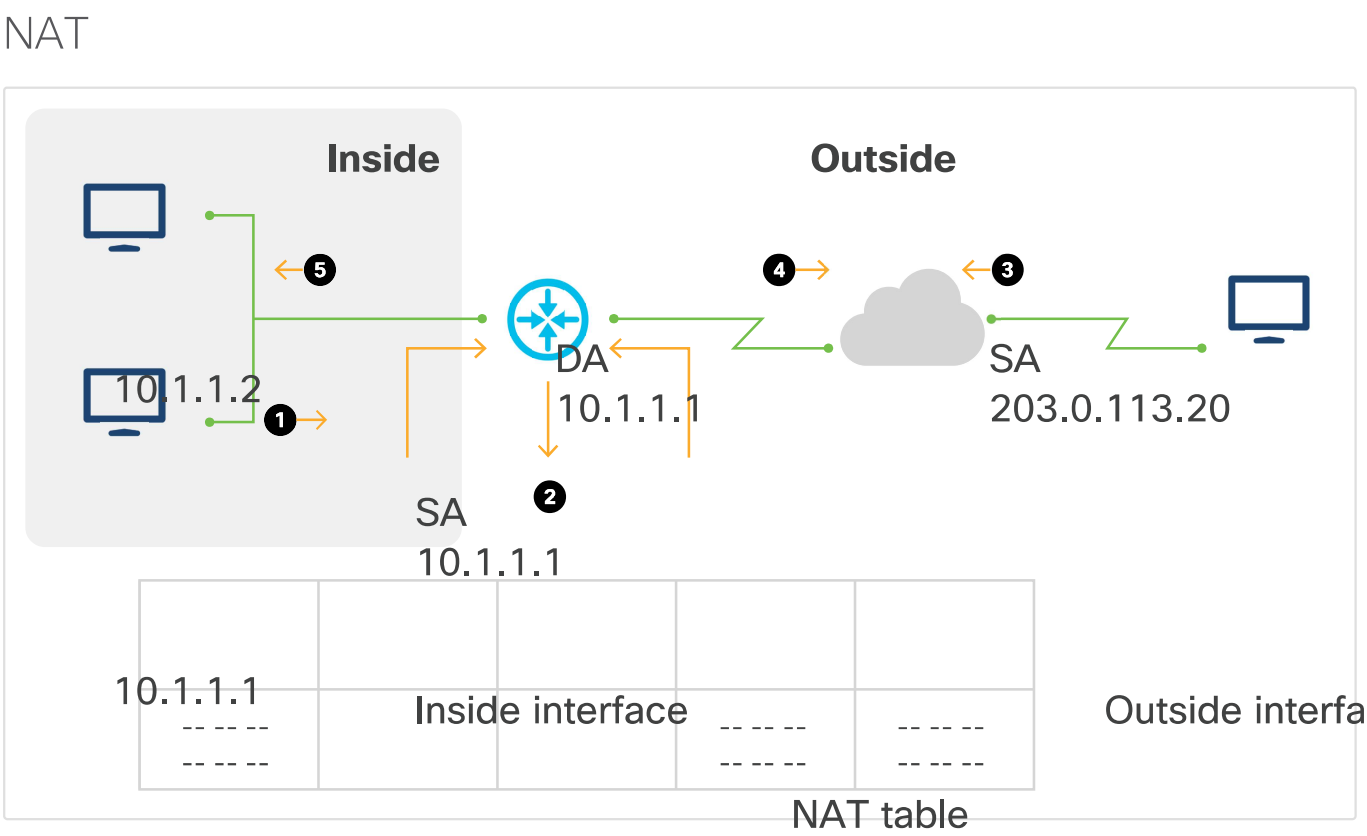
- **Local address** – This is any address that appears on the inside portion of the network.
- **Global address** – This is any address that appears on the outside portion of the network.

Inside Source Address Translation

IPv4 addresses can be translated into globally-unique IPv4 addresses when communicating outside the internal network. There are two options to accomplish this:

- **Static translation** - This method sets up a one-to-one mapping between an inside local address and an inside global address. This is useful when a host on the inside must be accessed from a fixed outside address.
- **Dynamic translation** - This method maps between inside local addresses and a global address pool.

The figure shows a device translating a source address inside a network to a source address outside the network:



As shown in the above figure, inside source address translation works as follows:

1. The user at 10.1.1.1 initiates a connection to Host B in the outside network.
2. When a packet is received from 10.1.1.1, the NAT device checks its NAT table and determines what to do next:

- If a static mapping is found, the device goes to Step 3, below.

Protocol	Inside Local IP Address	Inside Global IP Address	Outside Global IP Address
---	10.1.1.2	203.0.113.30	---

The NAT device swaps the inside local source address of host 10.1.1.1 with the global address of the translation entry, then forwards the packet.

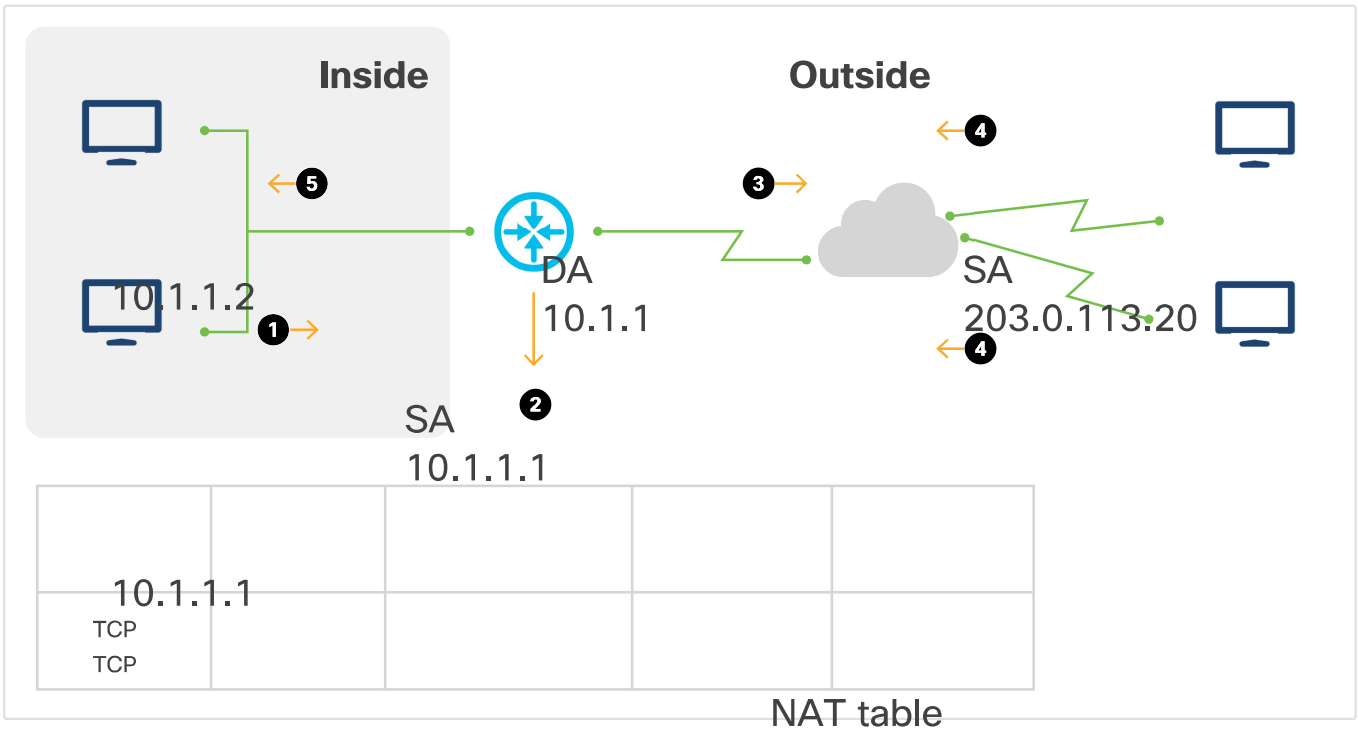
3. Host B receives the packet and replies to host 10.1.1.1 using the inside global IP destination address (DA) 203.0.113.20.

4. The NAT device uses the inside global address as a key, performs a NAT table lookup, and then translates it to the inside local address of host 10.1.1.1 before forwarding the packet.
5. Host 10.1.1.1 receives the packet and continues the exchange, beginning with Step 1 above.

Overloading of Inside Global Addresses

Using a single global address for multiple local addresses is known as overloading. When overloading is configured, the NAT device gathers information from higher-level protocols (for example, TCP or UDP port numbers) to translate global addresses back to correct local addresses. To map multiple local addresses to one global address, TCP or UDP port numbers are used to distinguish local addresses. This NAT process is called Port Address Translation (PAT). An example is shown in the following figure.

PAT



PAT works as follows. Both Host B and Host C think they are communicating with a single host at address 203.0.113.20. They are actually communicating with different hosts, differentiated by port number:

Protocol	Inside Local IP Address:port	Inside Global IP Address:port	Outside Global IP Address:port
TCP	10.1.1.1:1024	203.0.113.20:1024	198.51.100.4:23

1. Host 10.1.1.1 initiates a connection to Host B.
2. The first packet received from host 10.1.1.1 causes the device to check its NAT table:
   - If no translation entry exists, the device translates the inside local address to a global address from the available pool.
   - If another translation is ongoing (presuming overloading is enabled), the device reuses that translation's global address and saves information in the NAT table that can be used to reverse the process, translating the global address back to the proper local address. This is called an extended entry.

     The NAT device swaps source address 10.1.1.1 with the global address, then forwards the packet.
3. Host B receives the packet and responds by using the inside global IP address 203.0.113.20.

4. The NAT device then uses the inside and outside addresses and port numbers to perfors a NAT table lookup. It translates to the inside local address 10.1.1.1 and forwards the packet.
5. Host 10.1.1.1 receives the packet and continues the exchange, beginning with Step 1 above.

5.5.7

Packet Tracer – Explore Network Protocols

Many services run on networks behind the scene to make things happen reliably and efficiently. As a developer, you should understand what services are available and how they can help you. You should also understand the basics of how the most useful and popular services are configured. In Packet Tracer, these services are simulated, and the configuration is simple and straightforward. However, Packet Tracer does a very good job at simulating the actual traffic. As you work through this lab and send traffic, we encourage you to switch to Simulation mode to explore the contents of the various types of packets that the network is generating.

You will complete these objectives:

- Part 1: Configure DNS
- Part 2: Configure DHCP
- Part 3: Configure NTP
- Part 4: Use SSH to Configure a Switch
- Part 5: Use SNMP
- Part 6: Configure HTTPS
- Part 7: Configure EMAIL
- Part 8: Configure FTP

Explore Network Protocols

Explore Network Protocols

5.4
Network Devices

5.6
Troubleshooting Application Connectivity I...