IP Addressing: DNS Configuration Guide, Cisco IOS Release 15SY

Americas Headquarters
Cisco Systems, Inc.
170 West Tasman Drive
San Jose, CA 95134-1706
USA
http://www.cisco.com
Tel: 408 526-4000
800 553-NETS (6387)
Fax: 408 527-0883

CONTENTS

CHAPTER 1 **Configuring DNS** 1

- Finding Feature Information 1
- Prerequisites for Configuring DNS 2
- Information About DNS 2
  - DNS Overview 2
    - Hostnames for Network Devices 2
    - Domains Names for Groups of Networks 2
    - Name Servers 2
    - Cache 2
    - Name Resolvers 3
    - Zones 3
    - Authoritative Name Servers 3
    - DNS Operation 3
- How to Configure DNS 4
  - Mapping Hostnames to IP Addresses 4
  - Customizing DNS 5
  - Configuring DNS Spoofing 7
  - Configuring the Device as a DNS Server 8
    - Examples 11
      - Debugging Output for Relaying a DNS Query to Another Name Server Example 11
      - Debugging Output for Servicing a DNS Query from the Local Host Table Example 11
  - Disabling DNS Queries for ISO CLNS Addresses 11
  - Verifying DNS 12
- Configuration Examples for DNS 13
  - Example: IP Addresses 13
  - Example: Mapping Hostnames to IP Addresses 13
  - Example: Customizing DNS 14

Example: Configuring DNS Spoofing 14
Additional References 14
Feature Information for DNS 15

CHAPTER 2 Dynamic DNS Support for Cisco IOS Software 17
Finding Feature Information 17
Restrictions for Dynamic DNS Support for Cisco IOS Software 17
Information About Dynamic DNS Support for Cisco IOS Software 18
Domain Name System and Dynamic Updates 18
DDNS Updates for HTTP-Based Protocols 18
DHCP Support for DDNS Updates 18
Feature Design of Dynamic DNS Support for Cisco IOS Software 19
How to Configure Dynamic DNS Support for Cisco IOS Software 19
Configuring a Host List 20
Verifying the Host-List Configuration 21
Configuring DHCP Support of DDNS Updates 24
Configuring DDNS Update Support on Interfaces 26
Configuring a Pool of DHCP Servers to Support DDNS Updates 28
Configuring the Update Method and Interval 30
Verifying DDNS Updates 34
Configuration Examples for Dynamic DNS Support for Cisco IOS Software 39
Configuration of the DHCP Client Example 39
Configuration of the DHCP Server Example 39
Configuration of the HTTP Updates Example 39
Additional References 41
Feature Information for Dynamic DNS Support for Cisco IOS Software 43

CHAPTER 3 VRF-Aware DNS 45
Finding Feature Information 45
Information About VRF-Aware DNS 46
Domain Name System 46
VRF Mapping and VRF-Aware DNS 46
How to Configure VRF-Aware DNS 47
Defining a VRF Table and Assigning a Name Server to Enable VRF-Aware DNS 47
Mapping VRF-Specific Hostnames to IP Addresses 48

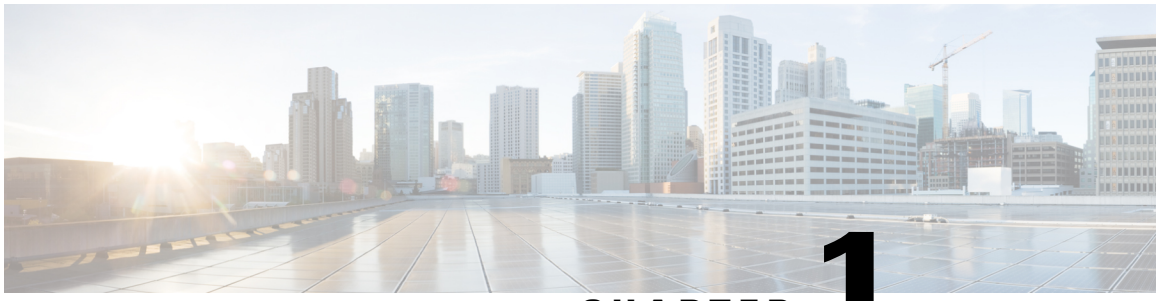
Configuring a Static Entry in a VRF-Specific Name Cache 49
Verifying the Name Cache Entries in the VRF Table 50
Configuration Examples for VRF-Aware DNS 51
Example: VRF-Specific Name Server Configuration 51
Example: VRF-Specific Domain Name List Configuration 51
Example: VRF-Specific Domain Name Configuration 52
Example: VRF-Specific IP Host Configuration 52
Additional References 52
Feature Information for VRF-Aware DNS 53

CHAPTER 4 **Split DNS** 55
Finding Feature Information 55
Prerequisites for Split DNS 56
Restrictions for Split DNS 56
Information About Split DNS 56
Split DNS Feature Overview 56
Split DNS Use to Respond to DNS Queries Benefits 56
Selection of Virtual DNS Caching Name Server Configurations 56
Ability to Offload Internet Traffic from the Corporate DNS Server 57
Compatibility with NAT and PAT 57
Split DNS Operation 58
CPE Device Configuration 58
DNS Query Issued by a CPE Client 59
Virtual DNS Name Server Selection 59
Response to the Client-issued DNS Query 59
DNS Views 60
View Use Is Restricted to Queries from the Associated VRF 60
Parameters for Resolving Internally Generated DNS Queries 61
Parameters for Forwarding Incoming DNS Queries 61
DNS View Lists 62
DNS Name Groups 63
DNS View Groups 64
Device Response to DNS Queries in a Split DNS Environment 64
Response to Incoming DNS Queries per the Forwarding Parameters of the Selected DNS View 65

Response to Internally Generated DNS Queries per the Resolving Parameters of the Default Global DNS View 66
How to Configure Split DNS 66
Enabling Split DNS Debugging Output 66
Defining a DNS Name List 68
Defining a DNS View 69
Defining Static Entries in the Hostname Cache for a DNS View 73
Defining a DNS View List 75
Modifying a DNS View List 77
Adding a Member to a DNS View List Already in Use 77
Changing the Order of the Members of a DNS View List Already in Use 78
Specifying the Default DNS View List for the DNS Server of the Device 80
Specifying a DNS View List for a Device Interface 81
Specifying a Source Interface to Forward DNS Queries 82
Configuration Examples for Split DNS 84
Example: Split DNS View Limited to Queries from a Specific VRF 84
Example: Split DNS View with Dynamic Name Server Configuration 84
Example: Split DNS View with Statically Configured Hostname Cache Entries 85
Example: Split DNS View with Round-Robin Rotation of Hostname Cache Entries 85
Example: Split DNS Configuration of ACLs That Can Limit DNS View Use 85
Example: Split DNS View Lists Configured with Different View-use Restrictions 86
Example: Split DNS Configuration of Default and Interface-specific View Lists 87
Additional References 88
Feature Information for Split DNS 89
Glossary 89

CHAPTER 5 **Service Discovery Gateway** 93
Information About Service Discovery Gateway 93
Service Announcement Redistribution and Service Extension 93
Extending Services Across Subnets—An Overview 94
Set Filter Options to Extend Services Across Subnets 95
Extend Services Across Subnets 97
How to Configure Service Discovery Gateway 99
Setting Filter Options for Service Discovery 99
Applying Service Discovery Filters and Configuring Service Discovery Parameters 101

Applying Service Discovery Filters for an Interface 103
Creating a Service Instance 105
Verifying and troubleshooting Service Discovery Gateway 106
Configuration Examples for Service Discovery Gateway 108
Example: Setting Filter Options for Service Discovery 108
Example: Applying Service Discovery Filters and Configuring Service Discovery Parameters 108
Example: Applying Service Discovery Filters for an Interface 109
Example: Setting Multiple Service Discovery Filter Options 109
Example: Creating a Service Instance 111
Additional References for Service Discovery Gateway 111
Feature Information for Service Discovery Gateway 112



CHAPTER 1

Configuring DNS

The Domain Name System (DNS) is a distributed database in which you can map hostnames to IP addresses through the DNS protocol from a DNS server. Each unique IP address can have an associated hostname. The Cisco IOS software maintains a cache of hostname-to-address mappings for use by the **connect**, **telnet**, and **ping** EXEC commands, and related Telnet support operations. This cache speeds the process of converting names to addresses.

Note You can specify IPv4 and IPv6 addresses while performing various tasks in this feature. The resource record type AAAA is used to map a domain name to an IPv6 address. The IP6.ARPA domain is defined to look up a record given an IPv6 address.

- Finding Feature Information, page 1
- Prerequisites for Configuring DNS, page 2
- Information About DNS, page 2
- How to Configure DNS, page 4
- Configuration Examples for DNS, page 13
- Additional References, page 14
- Feature Information for DNS, page 15

Finding Feature Information

Your software release may not support all the features documented in this module. For the latest caveats and feature information, see Bug Search Tool and the release notes for your platform and software release. To find information about the features documented in this module, and to see a list of the releases in which each feature is supported, see the feature information table at the end of this module.

Use Cisco Feature Navigator to find information about platform support and Cisco software image support. To access Cisco Feature Navigator, go to www.cisco.com/go/cfn. An account on Cisco.com is not required.

Prerequisites for Configuring DNS

To use DNS, you must have a DNS name server on your network.

Information About DNS

DNS Overview

If your network devices require connectivity with devices in networks for which you do not control name assignment, you can assign device names that uniquely identify your devices within the entire internetwork. The global naming scheme of the Internet, the DNS, accomplishes this task. This service is enabled by default. The following sections summarize DNS concepts and function.

Hostnames for Network Devices

Each unique IP address can have an associated hostname. DNS uses a hierarchical scheme for establishing hostnames for network nodes. This allows local control of the segments of the network through a client-server scheme. The DNS system can locate a network device by translating the hostname of the device into its associated IP address.

Domains Names for Groups of Networks

IP defines a naming scheme that allows a device to be identified by its location in the IP. This is a hierarchical naming scheme that provides for *domains*. On the Internet, a domain is a portion of the naming hierarchy tree that refers to general groupings of networks based on organization type or geography. Domain names are pieced together with periods (.) as the delimiting characters. For example, Cisco is a commercial organization that the IP identifies by a *com* domain name, so its domain name is *cisco.com*. A specific device in this domain, the File Transfer Protocol (FTP) system, for example, is identified as *ftp.cisco.com*.

Name Servers

To keep track of domain names, IP has defined the concept of a *name server*. Name servers are programs that have complete information about their namespace portion of the domain tree and may also contain pointers to other name servers that can be used to lead to information from any other part of the domain tree. Name servers know the parts of the domain tree for which they have complete information. A name server may also store information about other parts of the domain tree. Before domain names can be mapped to IP addresses, you must first identify the hostnames, then specify a name server, and enable the DNS service.

Cache

To speed the process of converting names to addresses, the name server maintains a database, called a *cache*, of hostname-to-address mappings for use by the **connect**, **telnet**, and **ping** EXEC commands, and related Telnet support operations. The cache stores the results from previous responses. Upon receiving a client-issued DNS query, the name server will check this local storage to see if the answer is available locally.

Name Resolvers

Name resolvers are programs that extract information from name servers in response to client requests. Resolvers must be able to access at least one name server. The resolver either uses that name server's information to answer a query directly or pursues the query using referrals to other names servers. A resolver will typically be a system routine that is directly accessible to user programs. Therefore, no protocol is necessary between the resolver and the user program.

Zones

The domain namespace is divided into areas called zones that are points of delegation in the DNS tree. A zone contains all domains from a certain point downward, except those for which other zones are authoritative.

Authoritative Name Servers

A name server is said to be an authority for the parts of the domain tree for which it has complete information. A zone usually has an authoritative name server, often more than one. An authoritative name server has been configured with host table information or has acquired host table information though a zone transfer (the action that occurs when a secondary DNS server starts up and updates itself from the primary server).

DNS Operation

An organization can have many name servers, but Internet clients can query only those that the root name servers know. The other name servers answer internal queries only.

A name server handles client-issued queries to the DNS server for locally defined hosts within a particular zone as follows:

- An authoritative name server responds to DNS user queries for a domain name that is under its zone of authority by using the permanent and cached entries in its own host table. If the query is for a domain name that is under its zone of authority but for which it does not have any configuration information, the authoritative name server simply replies that no such information exists.
- A name server that is not configured as the authoritative name server responds to DNS user queries by using information that it has cached from previously received query responses. If no device is configured as the authoritative name server for a zone, queries to the DNS server for locally defined hosts will receive nonauthoritative responses.

Name servers answer DNS queries (forward incoming DNS queries or resolve internally generated DNS queries) according to the forwarding and lookup parameters configured for the specific domain.

When DNS queries are forwarded to name servers for resolution, some memory space is held for the corresponding DNS query until an appropriate response is received or until there is timeout. To avoid the free I/O memory from getting exhausted when handling queries at high rate, configure the maximum size for the queue.

How to Configure DNS

Mapping Hostnames to IP Addresses

Perform this task to map hostnames to IP addresses.

A name server is used to keep track of information associated with domain names. A name server can maintain a database of hostname-to-address mappings. Each name can map to one or more IP addresses. In order to use this service to map domain names to IP addresses, you must specify a name server.

The name lookup system can be statically configured using the commands described in this task. Some other functions in Cisco IOS software, such as DHCP, can dynamically modify the state of the name lookup system. Use the **show hosts** command to display the cached hostnames and the DNS configuration.

SUMMARY STEPS

1. **enable**
2. **configure terminal**
3. **ip host** *name* [*tcp-port-number*] *address1* [*address2 ... address8*]
4. Do one of the following:
   - **ip domain name** *name*
   - **ip domain list** *name*
5. **ip name-server** *server-address1* [*server-address2 ... server-address6*]
6. **ip domain lookup** [**source-interface** *interface-type interface-number*]

DETAILED STEPS

	Command or Action	Purpose
Step 1	**enable** **Example:** `Device> enable`	Enables privileged EXEC mode. • Enter your password if prompted.
Step 2	**configure terminal** **Example:** `Device# configure terminal`	Enters global configuration mode.
Step 3	**ip host** *name* [*tcp-port-number*] *address1* [*address2 ... address8*]	Defines a static hostname-to-address mapping in the hostname cache. • The host IP address can be an IPv4 or IPv6 address.

	Command or Action	Purpose
	Example: `Device(config)# ip host cisco-rtp 192.168.0.148`	• Typically, it is easier to refer to network devices by symbolic names rather than numerical addresses (services such as Telnet can use hostnames or addresses). Hostnames and IP addresses can be associated with one another through static or dynamic means. • Manually assigning hostnames to addresses is useful when dynamic mapping is not available.

Step 4	Do one of the following:
	• **ip domain name** *name*
	• **ip domain list** *name*
	Example:
	`Device(config)# ip domain name cisco.com`
	Example:
	Example:
	`Device(config)# ip domain list cisco1.com`
		(Optional) Defines a default domain name that the Cisco IOS software will use to complete unqualified hostnames.
		or
		(Optional) Defines a list of default domain names to complete unqualified hostnames.
		• You can specify a default domain name that the Cisco IOS software will use to complete domain name requests. You can specify either a single domain name or a list of domain names. Any hostname that does not contain a complete domain name will have the default domain name you specify appended to it before the name is looked up.
		Note If there is no domain list, the domain name that you specified with the **ip domain name** global configuration command is used. If there is a domain list, the default domain name is not used. The **ip domain list** command is similar to the **ip domain name** command, except that with the **ip domain list** command you can define a list of domains, each to be tried in turn until the system finds a match.

Step 5	**ip name-server** *server-address1* [*server-address2 ... server-address6*] **Example:** `Device(config)# ip name-server 172.16.1.111 172.16.1.2`	Specifies one or more hosts (up to six) that can function as a name server to supply name information for DNS.
Step 6	**ip domain lookup** [**source-interface** *interface-type interface-number*] **Example:** `Device(config)# ip domain lookup`	(Optional) Enables DNS-based address translation. • DNS is enabled by default. Use this command if DNS has been disabled.

Customizing DNS

Perform this task to customize your DNS configuration.

In a multiple server configuration without the DNS round-robin functionality, many programs will use the first host server/IP address for the whole time to live (TTL) of the cache and use the second and third host servers/IP addresses only in the event of host failure. This behavior presents a problem when a high volume of users all arrive at the first host during the TTL time. For example, the network access server (NAS) sends out a DNS query. The DNS servers reply with a list of the configured IP addresses to the NAS. The NAS then caches these IP addresses for a given time (for example, five minutes). All users that dial in during the five minute TTL time will land on one host, the first IP address in the list.

In a multiple server configuration with the DNS round-robin functionality, the DNS server returns the IP address of all hosts to rotate between the cache of hostnames. During the TTL of the cache, users are distributed among the hosts. This functionality distributes calls across the configured hosts and reduces the number of DNS queries.

In a scheduling algorithm, processes are activated in a fixed cyclic order. Processes that are waiting for other events, like termination of a child process or an input or output operation, cannot proceed and hence they return control to the scheduler. If the TTL of the process times out just before the event (for which it was waiting) occurs, then the event will not be handled until all the other processes are activated.

Note The DNS round-robin functionality is applicable only for the DNS lookups on a device and is not applicable to another client pointing to the device.

SUMMARY STEPS

1. **enable**
2. **configure terminal**
3. **ip domain timeout** *seconds*
4. **ip domain retry** *number*
5. **ip domain round-robin**

DETAILED STEPS

	Command or Action	Purpose
Step 1	**enable** **Example:** `Device> enable`	Enables privileged EXEC mode. • Enter your password if prompted.
Step 2	**configure terminal** **Example:** `Device# configure terminal`	Enters global configuration mode.
Step 3	**ip domain timeout** *seconds* **Example:** `Device(config)# ip domain timeout 17`	(Optional) Specifies the amount of time to wait for a response to a DNS query.

	Command or Action	Purpose
		• If the **ip domain timeout** command is not configured, the Cisco IOS software will wait 3 seconds for a response to a DNS query.
Step 4	**ip domain retry** *number* **Example:** `Device(config)# ip domain retry 10`	(Optional) Specifies the number of times to retry sending DNS queries. • If the **ip domain retry** command is not configured, the Cisco IOS software will retry DNS queries twice.
Step 5	**ip domain round-robin** **Example:** `Device(config)# ip domain round-robin`	(Optional) Enables round-robin functionality on DNS servers.

Configuring DNS Spoofing

Perform this task to configure DNS spoofing.

DNS spoofing is designed to allow a device to act as a proxy DNS server and "spoof" replies to any DNS queries using either the configured IP address in the **ip dns spoofing** *ip-address* command or the IP address of the incoming interface for the query. This feature is useful for devices where the interface toward the Internet service provider (ISP) is not up. Once the interface to the ISP is up, the device forwards DNS queries to the real DNS servers.

This feature turns on DNS spoofing and is functional if any of the following conditions are true:

- The **no ip domain lookup** command is configured.
- IP name server addresses are not configured.
- There are no valid interfaces or routes for sending to the configured name server addresses.

If these conditions are removed, DNS spoofing will not occur.

SUMMARY STEPS

1. **enable**
2. **configure terminal**
3. **ip dns server**
4. **ip dns spoofing** [*ip-address*]

DETAILED STEPS

	Command or Action	Purpose
Step 1	**enable** **Example:** `Device> enable`	Enables privileged EXEC mode. • Enter your password if prompted.
Step 2	**configure terminal** **Example:** `Device# configure terminal`	Enters global configuration mode.
Step 3	**ip dns server** **Example:** `Device(config)# ip dns server`	Activates the DNS server on the device.
Step 4	**ip dns spoofing** [*ip-address*] **Example:** `Device(config)# ip dns spoofing 192.168.15.1`	Configures DNS spoofing. • The IP address used for DNS spoofing can be an IPv4 or IPv6 address. • The device will respond to the DNS query with the configured *ip-address* when queried for any hostname other than its own. • The device will respond to the DNS query with the IP address of the incoming interface when queried for its own hostname.

Configuring the Device as a DNS Server

Perform this task to configure the device as a DNS server.

A Cisco IOS device can provide service to DNS clients, acting as both a caching name server and as an authoritative name server for its own local host table.

When configured as a caching name server, the device relays DNS requests to other name servers that resolve network names into network addresses. The caching name server caches information learned from other name servers so that it can answer requests quickly, without having to query other servers for each transaction.

When configured as an authoritative name server for its own local host table, the device listens on port 53 for DNS queries and then answers DNS queries using the permanent and cached entries in its own host table.

An authoritative name server usually issues zone transfers or responds to zone transfer requests from other authoritative name servers for the same zone. However, the Cisco IOS DNS server does not perform zone transfers.

When it receives a DNS query, an authoritative name server handles the query as follows:

- If the query is for a domain name that is not under its zone of authority, the authoritative name server determines whether to forward the query to specific back-end name servers based on whether IP DNS-based hostname-to-address translation has been enabled via the **ip domain lookup** command.
- If the query is for a domain name that is under its zone of authority and for which it has configuration information, the authoritative name server answers the query using the permanent and cached entries in its own host table.
- If the query is for a domain name that is under its zone of authority but for which it does not have any configuration information, the authoritative name server does not forward the query elsewhere for a response; instead the authoritative name server simply replies that no such information exists.

Note Unless Distributed Director is enabled, the TTL on locally defined resource records will always be ten seconds, regardless of any authority record parameters that may have been specified for the DNS name server by the use of the **ip dns primary** command.

SUMMARY STEPS

1. **enable**
2. **configure terminal**
3. **ip dns server**
4. **ip name-server** *server-address1* [*server-address2... server-address6*]
5. **ip dns server queue limit** {**forwarder** *queue-size-limit* | **director** *queue-size-limit*}
6. **ip host** [**vrf** *vrf-name*] [**view** *view-name*] *hostname* {*address1* [*address2 ... address8*] | **additional** *address9* [*address10 ... addressn*]}
7. **ip dns primary** *domain-name* **soa** *primary-server-name mailbox-name* [*refresh-interval* [*retry-interval* [*expire-ttl* [*minimum-ttl*]]]]
8. **ip host** *domain-name* **ns** *server-name*

DETAILED STEPS

	Command or Action	Purpose
Step 1	**enable** **Example:** `Device> enable`	Enables privileged EXEC mode. • Enter your password if prompted.
Step 2	**configure terminal** **Example:** `Device# configure terminal`	Enters global configuration mode.

	Command or Action	Purpose
Step 3	**ip dns server** **Example:** `Device(config)# ip dns server`	Enables the DNS server.
Step 4	**ip name-server** *server-address1* [*server-address2... server-address6*] **Example:** `Device(config)# ip name-server 192.168.2.120 192.168.2.121`	(Optional) Configures other DNS servers: • Cisco IOS resolver name servers • DNS server forwarders **Note** If the Cisco IOS name server is being configured to respond only to domain names for which it is authoritative, there is no need to configure other DNS servers.
Step 5	**ip dns server queue limit** {**forwarder** *queue-size-limit* | **director** *queue-size-limit*} **Example:** `Device(config)# ip dns server queue limit forwarder 10`	(Optional) Configures a limit to the size of the queues used by the DNS server processes. • The **director** keyword was removed in Cisco IOS Release 12.4(24)T.
Step 6	**ip host** [**vrf** *vrf-name*] [**view** *view-name*] *hostname* {*address1* [*address2 ... address8*] | **additional** *address9* [*address10 ... addressn*]} **Example:** `Device(config)# ip host user1.example.com 192.168.201.5 192.168.201.6`	(Optional) Configures local hosts.
Step 7	**ip dns primary** *domain-name* **soa** *primary-server-name mailbox-name* [*refresh-interval* [*retry-interval* [*expire-ttl* [*minimum-ttl*]]]] **Example:** `Device(config)# ip dns primary example.com soa ns1.example.com mb1.example.com`	Configures the device as the primary DNS name server for a domain (zone) and as the start of authority (SOA) record source (which designates the start of a zone). **Note** Unless Distributed Director is enabled, the TTL on locally defined resource records will always be ten seconds.
Step 8	**ip host** *domain-name* **ns** *server-name* **Example:** `Device(config)# ip host example.com ns ns1.example.com`	(Optional) Configures the device to create an name server (NS) resource record to be returned when the DNS server is queried for the associated domain. • This configuration is needed only if the zone for which the system is authoritative will also be served by other name servers.

Examples

This section provides examples of debugging output that is logged when a device is configured as an authoritative name server for its own local host table and the **debug domain** command is in effect:

Note For DNS-based X.25 routing, the **debug x25 events** command supports functionality to describe the events that occur while the X.25 address is being resolved to an IP address using a DNS server. The **debug domain** command can be used along with **debug x25 events** to observe the whole DNS-based X.25 routing data flow.

Debugging Output for Relaying a DNS Query to Another Name Server Example

The following is sample output from the **debug domain** command that corresponds to relaying a DNS query to another name server when the device is configured as an authoritative name server for its own local host table:

```
Apr  4 22:18:32.183: DNS: Incoming UDP query (id#18713)
Apr  4 22:18:32.183: DNS: Type 1 DNS query (id#18713) for host 'ns1.example.com' from
192.0.2.120(1283)
Apr  4 22:18:32.183: DNS: Re-sending DNS query (type 1, id#18713) to 192.0.2.121
Apr  4 22:18:32.211: DNS: Incoming UDP query (id#18713)
Apr  4 22:18:32.211: DNS: Type 1 response (id#18713) for host <ns1.example.com> from
192.0.2.121(53)
Apr  4 22:18:32.215: DOM: dom2cache: hostname is ns1.example.com, RR type=1, class=1,
ttl=86400, n=4
Apr  4 22:18:32.215: DNS: Forwarding back A response - no director required
Apr  4 22:18:32.215: DNS: Finished processing query (id#18713) in 0.032 secs
Apr  4 22:18:32.215: DNS: Forwarding back reply to 192.0.2.120/1283
```

Debugging Output for Servicing a DNS Query from the Local Host Table Example

The following is sample output from the **debug domain** command that corresponds to servicing a DNS query from the local host table when the device is configured as an authoritative name server for its own local host table:

```
Apr  4 22:16:35.279: DNS: Incoming UDP query (id#8409)
Apr  4 22:16:35.279: DNS: Type 1 DNS query (id#8409) for host 'ns1.example.com' from
192.0.2.120(1279)
Apr  4 22:16:35.279: DNS: Finished processing query (id#8409) in 0.000 secs
```

Disabling DNS Queries for ISO CLNS Addresses

Perform this task to disable DNS queries for International Organization for Standardization (ISO) Connectionless Network Service (CLNS) addresses.

If your device has both IP and ISO CLNS enabled and you want to use ISO CLNS network service access point (NSAP) addresses, you can use the DNS to query these addresses, as documented in RFC 1348. This feature is enabled by default.

SUMMARY STEPS

1. **enable**
2. **configure terminal**
3. **no ip domain lookup nsap**

DETAILED STEPS

	Command or Action	Purpose
Step 1	**enable** **Example:** `Device> enable`	Enables privileged EXEC mode. • Enter your password if prompted.
Step 2	**configure terminal** **Example:** `Device# configure terminal`	Enters global configuration mode.
Step 3	**no ip domain lookup nsap** **Example:** `Device(config)# no ip domain lookup nsap`	Disables DNS queries for ISO CLNS addresses.

Verifying DNS

Perform this task to verify your DNS configuration.

1 enable

2 ping *hosts*

3 show hosts

SUMMARY STEPS

1. **enable**
2. **ping** *hosts*
3. **show hosts**

DETAILED STEPS

	Command or Action	Purpose
Step 1	**enable** **Example:** `Device> enable`	Enables privileged EXEC mode. • Enter your password if prompted.
Step 2	**ping** *hosts* **Example:** `Device# ping cisco-rtp`	Diagnoses basic network connectivity. • After the DNS configuration is set, you can verify the DNS server by using a hostname to ping or telnet to a device.
Step 3	**show hosts** **Example:** `Device# show hosts`	Displays the default domain name, the style of name lookup service, a list of name server hosts, and the cached list of hostnames and addresses. • After a name is resolved using DNS, use the **show hosts** command to view the cached hostnames and the DNS configuration.

Configuration Examples for DNS

Example: IP Addresses

The following example establishes a domain list with several alternate domain names:

```
ip domain list example.com
ip domain list example1.edu
ip domain list example2.edu
```

Example: Mapping Hostnames to IP Addresses

The following example configures the hostname-to-address mapping process. IP DNS-based translation is specified, the addresses of the name servers are specified, and the default domain name is given.

```
! IP DNS-based hostname-to-address translation is enabled
ip domain lookup
! Specifies hosts 192.168.1.111 and 192.168.1.2 as name servers
ip name-server 192.168.1.111 192.168.1.2
! Defines cisco.com as the default domain name the device uses to complete
! Set the name for unqualified hostnames
ip domain name cisco.com
```

Example: Customizing DNS

The following example allows a Telnet to company.example.com to connect to each of the three IP addresses specified in the following order: the first time the hostname is referenced, it would connect to 10.0.0.1; the second time the hostname is referenced, it would connect to 10.1.0.1; and the third time the hostname is referenced, it would connect to 10.2.0.1. In each case, the other two addresses would also be tried if the first one failed; this is the normal operation of the Telnet command.

```
Device(config)# ip host company.example.com 10.0.0.1 10.1.0.1 10.2.0.1
Device(config)# ip domain round-robin
```

Example: Configuring DNS Spoofing

In the following example, the device is configured to spoof replies to any DNS queries:

```
ip dns server
ip dns spoofing
no ip domain lookup
interface e3/1
 ip address 10.1.1.1 255.255.255.0
```

Additional References

Related Documents

Related Topic	Document Title
DNS commands: complete command syntax, command mode, command history, defaults, usage guidelines, and examples	*Cisco IOS IP Addressing Services Command Reference*

Standards

Standards	Title
No new or modified standards are supported by this functionality.	--

MIBs

MIBs	MIBs Link
No new or modified MIBs are supported by this feature, and support for existing MIBs has not been modified by this feature.	To locate and download MIBs for selected platforms, Cisco IOS releases, and feature sets, use Cisco MIB Locator found at the following URL: http://www.cisco.com/go/mibs

RFCs

RFCs	Title
RFC 1348	*DNS NSAP RRs*

Technical Assistance

Description	Link
The Cisco Support website provides extensive online resources, including documentation and tools for troubleshooting and resolving technical issues with Cisco products and technologies. To receive security and technical information about your products, you can subscribe to various services, such as the Product Alert Tool (accessed from Field Notices), the Cisco Technical Services Newsletter, and Really Simple Syndication (RSS) Feeds. Access to most tools on the Cisco Support website requires a Cisco.com user ID and password.	http://www.cisco.com/techsupport

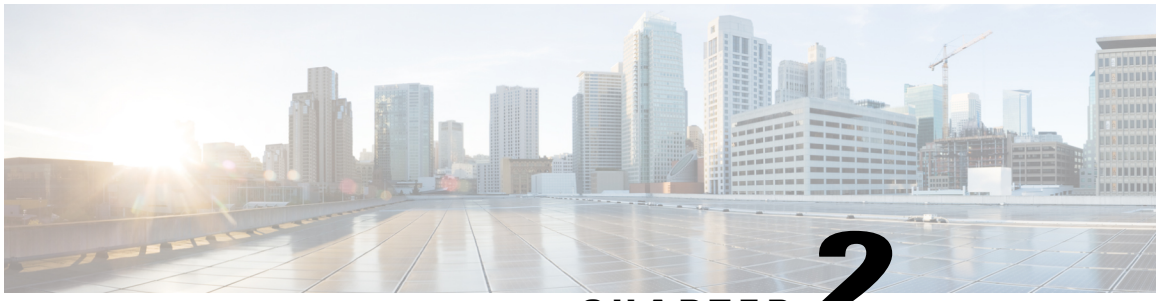
Feature Information for DNS

The following table provides release information about the feature or features described in this module. This table lists only the software release that introduced support for a given feature in a given software release train. Unless noted otherwise, subsequent releases of that software release train also support that feature.

Use Cisco Feature Navigator to find information about platform support and Cisco software image support. To access Cisco Feature Navigator, go to www.cisco.com/go/cfn. An account on Cisco.com is not required.

Table 1: Feature Information for DNS

Feature Name	Releases	Feature Information
DNS Spoofing	12.3(2)T 15.4(1)T	This feature is designed to allow a device to act as a proxy DNS server and "spoof" replies to any DNS queries using either the configured IP address in the **ip dns spoofing** *ip-address* command or the IP address of the incoming interface for the query. The following command was introduced by this feature: **ip dns spoofing**.



CHAPTER 2

Dynamic DNS Support for Cisco IOS Software

The Dynamic DNS Support for Cisco IOS Software feature enables Cisco IOS software devices to perform Dynamic Domain Name System (DDNS) updates to ensure that an IP host DNS name is correctly associated with its IP address.

It provides two mechanisms to generate or perform DDNS: the IETF standard as defined by RFC 2136 and a generic HTTP using various DNS services. With this feature, you can define a list of hostnames and IP addresses that will receive updates, specify an update method, and specify a configuration for Dynamic Host Configuration Protocol (DHCP) triggered updates.

- Finding Feature Information, page 17
- Restrictions for Dynamic DNS Support for Cisco IOS Software, page 17
- Information About Dynamic DNS Support for Cisco IOS Software, page 18
- How to Configure Dynamic DNS Support for Cisco IOS Software, page 19
- Configuration Examples for Dynamic DNS Support for Cisco IOS Software, page 39
- Additional References, page 41
- Feature Information for Dynamic DNS Support for Cisco IOS Software, page 43

Finding Feature Information

Your software release may not support all the features documented in this module. For the latest caveats and feature information, see Bug Search Tool and the release notes for your platform and software release. To find information about the features documented in this module, and to see a list of the releases in which each feature is supported, see the feature information table at the end of this module.

Use Cisco Feature Navigator to find information about platform support and Cisco software image support. To access Cisco Feature Navigator, go to www.cisco.com/go/cfn. An account on Cisco.com is not required.

Restrictions for Dynamic DNS Support for Cisco IOS Software

The performance of the DHCP client can be impacted when the Dynamic DNS Support for Cisco IOS Software feature is enabled, because of sending DDNS update packets and waiting for responses from the server (before

sending the ACK to the client REQUEST) and the client (immediately after receiving the ACK and assigning the address to the interface). The default for the client is two attempts with a 5-second wait time between attempts.

The DHCP server continues to process DHCP client DISCOVER and REQUEST packets while waiting for the DDNS updates to complete. Even if the update is done before sending the ACK to the client, it does not delay processing of other DHCP requests. The DHCP server could be impacted minimally because of the time and memory needed in order to set up the DDNS update and get things started.

Reloading the system may take a little longer in some cases, such as, if there are outstanding DDNS updates that need to complete.

Information About Dynamic DNS Support for Cisco IOS Software

Domain Name System and Dynamic Updates

The DNS was designed to support queries of a statically configured database. The data was expected to change, but minimally. All updates were made as external edits to a zone master file. The domain name identifies a node within the domain name space tree structure. Each node has a set (possibly empty) of Resource Records (RRs). All RRs having the same NAME, CLASS, and TYPE are called a Resource Record Set (RRset).

There are address (A) or forward RRs and pointer (PTR) or reverse RRs. The DDNS update can specify additions or deletions of hostnames and IP addresses. The two mechanisms to update this information are by using HTTP-based protocols such as DynDNS.org or by using the IETF standard.

DDNS Updates for HTTP-Based Protocols

The Dynamic DNS Support for Cisco IOS Software feature provides the capability of a proprietary HTTP-based protocol to generate or perform DDNS updates. The most notable HTTP-based protocol is DynDNS.org, but there are many others.

Since most of these protocols consist of a simple HTTP command that specifies parameters such as hostname and IP address in the URL portion of the command, this feature takes the same generic approach. You can specify the hostname and IP address in a URL. Configuration of a maximum interval between updates is also allowed.

DHCP Support for DDNS Updates

Before the Dynamic DNS Support for Cisco IOS Software feature, a DHCP server assigned IP addresses to DHCP clients and any DNS information was static. In a network that uses a DHCP server, there are many cases in which DNS hostnames should be associated with the IP addresses that are being assigned. There is an existing method for dynamically updating DNS for DHCP by using information in the fully qualified domain name (FQDN) DHCP option (if it is supplied by the client).

The Dynamic DNS Support for Cisco IOS Software feature enables the DHCP server to support a new FQDN DHCP option. In addition, when the address on an interface is configured, the client can pass the new FQDN option to the server so that name-to-address and address-to-name translations can be updated for the DHCP client as well.

Feature Design of Dynamic DNS Support for Cisco IOS Software

The Dynamic DNS Support for Cisco IOS Software feature enables the tracking of the FQDN DHCP option. If dynamic updates are enabled for the DHCP server, the server updates the PTR RR. The PTR RRs are used for reverse mapping (translation of addresses to names). PTRs use official names not aliases. The name in a PTR record is the local IP address portion of the reverse name.

If the client requests the server to update A RRs as well, the server will attempt to do it. The A RR provides the name-to-address mapping for a DNS zone. The server may be configured to override the client suggestion and always update PTR and A RRs.

The DHCP client can specify whether or not it wants to allow dynamic updates (include the FQDN option), instruct the server to allow the client to update both A and PTR RRs (normally only the A RR is updated by the client), and optionally instruct the server not to update any DNS information (either because the client will be updating both or simply because the client does not want the server to do any updates at all).

There are three basic components of the Dynamic DNS Support for Cisco IOS Software feature that are as follows:

- Definition of the hostname list and IP addresses that will receive updates using a new command that specifies a group of hostnames. Each configured list can consist of any number of IPv4 addresses or hostnames. If a hostname is configured, the name is translated to an IPv4 address at the time at which it is used.
- Specification of an update method. The options are HTTP, DDNS, or an internal Cisco IOS name cache. If the HTTP option is specified, the configuration will include a URL. The username and password must be explicitly written into the URL string and the entire “GET” operation must be specified on one line. The specification will be stored in a linked list. If the update method is DDNS, the configuration will include the update of the IP address.

Events that trigger updates can be as follows:

- IP address that is assigned by a DHCP server for an IP device
- IP address assigned to a router using a DHCP client
- Forwarding of the fully qualified domain name (FQDN) of a user or router hostname from the DHCP client to the server
- Point-to-Point Protocol (PPP)/IP Control Protocol (IPCP) obtaining an IP address for a router interface
- Forced update using a timer to verify a router IP address

Associated with each update method is a value specifying the maximum number of seconds between updates. If left unspecified, then the update is performed only when the address is changed. If specified, the update is performed automatically if the specified number of seconds have passed since the last update.

How to Configure Dynamic DNS Support for Cisco IOS Software

Note The internal Cisco IOS name cache does not require any configuration.

Configuring a Host List

Perform this task to configure a host list if you are going to use a host list in your configuration.

SUMMARY STEPS

1. **enable**
2. **configure terminal**
3. **ip host-list** *host-list-name*
4. **host** [**vrf** *vrf-name*] {*host-ip-address* | *hostname*}
5. **exit**

DETAILED STEPS

	Command or Action	Purpose
Step 1	**enable** **Example:** `Router> enable`	Enables privileged EXEC mode. • Enter your password if prompted.
Step 2	**configure terminal** **Example:** `Router# configure terminal`	Enters global configuration mode.
Step 3	**ip host-list** *host-list-name* **Example:** `Router(config)# ip host-list abc`	Specifies a list of hosts and enters host-list configuration mode. The *host-list-name argument*assigns a name to the list of hosts.
Step 4	**host** [**vrf** *vrf-name*] {*host-ip-address* | *hostname*} **Example:** `Router(host-list)# host 10.1.1.1 10.2.2.2 10.3.3.3 a.com b.com 10.4.4.4 10.5.5.5 d.com host 10.6.6.6 f.com host vrf abc a.com b.com c.com host vrf def 10.1.1.1 10.2.2.2 10.3.3.3`	Configures one or more hosts. The arguments and keyword are as follows: • **vrf** *vrf-name* --Associates a hostname with a virtual private network (VPN) routing and forwarding instance (VRF) name. **Note** All hostnames or IP addresses specified after the **vrf** keyword are associated with that VRF. • *host-ip-address* --Specifies an IP address for a host in the host list. You can specify more than one host using this argument by listing the hostname and IP addresses on the same line. • *hostname* --Specifies a hostname.

	Command or Action	Purpose
Step 5	**exit** **Example:** `Router(host-list)# exit`	Exits to global configuration mode.

Examples

The following example shows how to configure several hosts with VRF:

```
ip host-list abc
 host 10.1.1.1 10.2.2.2 10.3.3.3 a.com b.com 10.4.4.4 10.5.5.5 d.com
 host 10.6.6.6 f.com
 host vrf abc a.com b.com c.com
 host vrf def 10.1.1.1 10.2.2.2 10.3.3.3
```

Verifying the Host-List Configuration

To verify the host-list configuration, perform the following steps.

SUMMARY STEPS

1. **show ip host-list**
2. **show running-config | inc host-list**
3. **show running-config | inc host**
4. **debug ip ddns update**

DETAILED STEPS

Step 1 **show ip host-list**

Use this command to verify that the IP addresses and hostnames have been assigned to a host list, for example:

Example:

```
Router# show ip host-list abc
Host list: abc
 ddns.abc
 10.2.3.4
 ddns2.abc
 10.3.4.5
 ddns3.com
 10.3.3.3
 d.org
 e.org
 1.org.2.org
 3.com
 10.2.2.2 (VRF: test)
 10.5.5.5 (VRF: test)
```

```
 a.net (VRF: test)
 b.net (VRF: test)
```

Step 2 **show running-config | inc host-list**

Use this command to verify the configuration of a host list, for example:

Example:

```
Router# show running-config | inc host-list
ip host-list a
ip host-list b
ip host-list c
ip host-list abc
```

Step 3 **show running-config | inc host**

Use this command to verify the configuration of a hostname, for example:

Example:

```
Router# show running-config | inc host
hostname who
ip host who 10.0.0.2
ip host-list a
 host 10.1.1.1 a.com b.com 10.2.2.3 10.2.2.2 c.com. 10.3.3.3 10.4.4.4
 host d.com
 host vrf abc 10.10.10.4 10.10.10.8
 host vrf def 10.2.3.4 10.6.7.8
ip host-list b
 host a.com b.com c.com 10.1.1.1 10.2.2.2 10.3.3.3
 host vrf ppp 10.2.1.0
ip host-list c
 host 10.1.1.1 10.2.2.2 10.3.3.3 a.com b.com 10.4.4.4 10.5.5.5 d.com
 host 10.6.6.6 f.com
 host vrf zero a.com b.com c.om
 host vrf one 10.1.1.1 10.2.2.2 10.3.3.3
ip host-list unit-test
 host ddns.unit.test 10.2.3.4 ddns2.unit.test 10.3.4.5 ddns3.com 10.3.3.3 d.org e.org
 host 1.org.2.org 3.com
 host vrf ZERO 10.2.2.2 10.5.5.5 a.net b.net
 ip ddns update hostname use-this.host.name
 ip ddns update this-method host 10.2.3.4
 ip ddns update this-method host this-host
 ip ddns update this-method host-group this-list
 ip ddns update this-method host 10.3.4.5
 ip ddns update test host 10.19.192.32
 ip ddns update test host 10.19.192.32
 ip ddns update a host-group a
 ip ddns update a host-group ab
 ip ddns update aa host-group ab
 ip ddns update method host 10.33.44.55
```

Step 4 **debug ip ddns update**

Use the **debug ip ddns update** command for the following configuration to verify the configuration of the hosts. Two servers are configured in the host list. A DHCP client is configured for IETF DDNS updating of both A and DNS RRs and requesting the DHCP server to update neither. The DHCP client is configured to include an FQDN DHCP option that instructs the DHCP server not to update either A or PTR Resource Records. This is configured using the interface version of the command. The DHCP server is configured to allow the DHCP client to update whatever RRs it chooses.

Example:

```
!Configure the DHCP Client
ip host-list servers
```

```
 host 10.19.192.32 10.0.0.1
ip ddns update method testing
 ddns
interface Ethernet1
 ip dhcp client update dns server none
 ip ddns update testing host-group servers
 ip address dhcp
end
!Configure the DHCP Server
ip dhcp pool test
 network 10.0.0.0 255.0.0.0
 update dns
!Enable Debugging
debug ip ddns update
!The update to the server 10.0.0.1 fails in this example
00:18:58:%DHCP-6-ADDRESS_ASSIGN: Interface Ethernet1 assigned DHCP address 10.0.0.8, mask 255.0.0.0,
 hostname canada_reserved
00:18:58: DYNDNSUPD: Adding DNS mapping for canada_reserved.hacks <=> 10.0.0.8 server 10.19.192.32
00:18:58: DYNDNSUPD: Sleeping for 3 seconds waiting for interface Ethernet1 configuration to settle
00:19:01: DDNS: Enqueuing new DDNS update 'canada_reserved.hacks' <=> 10.0.0.8 server 10.19.192.32
00:19:01: DYNDNSUPD: Adding DNS mapping for canada_reserved.hacks <=> 10.0.0.8 server 10.0.0.1
00:19:01: DDNS: Enqueuing new DDNS update 'canada_reserved.hacks' <=> 10.0.0.8 server 10.0.0.1
00:19:01: DYNDNSUPD: Adding DNS mapping for canada_reserved.hacks <=> 10.0.0.8 server 10.0.0.1
00:19:01: DDNS: Enqueuing new DDNS update 'canada_reserved.hacks' <=> 10.0.0.8 server 10.0.0.1
00:19:01: DDNS: Zone name for '10.0.0.11.in-addr.arpa.' is '10.in-addr.arpa'
00:19:01: DDNS: Using server 10.19.192.32
00:19:01: DDNS: Dynamic Update 1: (sending to server 10.19.192.32)
00:19:01: DDNS: Zone = 10.in-addr.arpa
00:19:01: DDNS: Prerequisite: 10.0.0.11.in-addr.arpa. not in use
00:19:01: DDNS: Update: add 10.0.0.11.in-addr.arpa. IN PTR canada_reserved.hacks
00:19:01: DDNS: Zone name for '10.0.0.11.in-addr.arpa.' is '10.in-addr.arpa'
00:19:01: DDNS: Using server 10.0.0.1
00:19:01: DDNS: Dynamic Update 1: (sending to server 10.0.0.1)
00:19:01: DDNS: Zone = 10.in-addr.arpa
00:19:01: DDNS: Prerequisite: 10.0.0.11.in-addr.arpa. not in use
00:19:01: DDNS: Update: add 10.0.0.11.in-addr.arpa. IN PTR canada_reserved.hacks
00:19:01: DDNS: Zone name for '10.0.0.11.in-addr.arpa.' is '10.in-addr.arpa'
00:19:01: DDNS: Using server 10.0.0.1
00:19:01: DDNS: Dynamic Update 1: (sending to server 10.0.0.1)
00:19:01: DDNS: Zone = 10.in-addr.arpa
00:19:01: DDNS: Prerequisite: 10.0.0.11.in-addr.arpa. not in use
00:19:01: DDNS: Update: add 10.0.0.11.in-addr.arpa. IN PTR canada_reserved.hacks
00:19:01: DDNS: Dynamic DNS Update 1 (PTR) for host canada_reserved.hacks returned 6 (YXDOMAIN)
00:19:01: DDNS: Dynamic Update 2: (sending to server 10.19.192.32)
00:19:01: DDNS: Zone = 10.in-addr.arpa
00:19:01: DDNS: Update: delete 10.0.0.11.in-addr.arpa. all PTR RRs
00:19:01: DDNS: Update: add 10.0.0.11.in-addr.arpa. IN PTR canada_reserved.hacks
00:19:01: DDNS: Dynamic DNS Update 2 (PTR) for host canada_reserved.hacks returned 0 (NOERROR)
00:19:01: DDNS: Zone name for 'canada_reserved.hacks' is 'hacks'
00:19:01: DDNS: Using server 10.19.192.32
00:19:01: DDNS: Dynamic Update 1: (sending to server 10.19.192.32)
00:19:01: DDNS: Zone = hacks
00:19:01: DDNS: Prerequisite: canada_reserved.hacks not in use
00:19:01: DDNS: Update: add canada_reserved.hacks IN A 10.0.0.8
00:19:01: DDNS: Dynamic DNS Update 1 (A) for host canada_reserved.hacks returned 0 (NOERROR)
00:19:01: DDNS: Update of 'canada_reserved.hacks' <=> 10.0.0.8 finished
00:19:01: DYNDNSUPD: Another update completed (total outstanding=2)
00:19:11: DDNS: Dynamic DNS Update 1 (PTR) for host canada_reserved.hacks returned 0 (NOERROR)
00:19:11: DDNS: Dynamic DNS Update 1 (PTR) for host canada_reserved.hacks returned 0 (NOERROR)
00:19:11: DDNS: Zone name for 'canada_reserved.hacks' is 'hacks'
00:19:11: DDNS: Using server 10.0.0.1
00:19:11: DDNS: Dynamic Update 1: (sending to server 10.0.0.1)
00:19:11: DDNS: Zone = hacks
00:19:11: DDNS: Prerequisite: canada_reserved.hacks not in use
00:19:11: DDNS: Update: add canada_reserved.hacks IN A 10.0.0.8
00:19:11: DDNS: Zone name for 'canada_reserved.hacks' is 'hacks'
00:19:11: DDNS: Using server 10.0.0.1
00:19:11: DDNS: Dynamic Update 1: (sending to server 10.0.0.1)
00:19:11: DDNS: Zone = hacks
00:19:11: DDNS: Prerequisite: canada_reserved.hacks not in use
00:19:11: DDNS: Update: add canada_reserved.hacks IN A 10.0.0.8
```

```
00:19:21: DDNS: Dynamic DNS Update 1 (A) for host canada_reserved.hacks returned 0 (NOERROR)
00:19:21: DDNS: Update of 'canada_reserved.hacks' <=> 10.0.0.8 failed
00:19:21: DYNDNSUPD: Another update completed (total outstanding=1)
00:19:21: DDNS: Dynamic DNS Update 1 (A) for host canada_reserved.hacks returned 0 (NOERROR)
00:19:21: DDNS: Update of 'canada_reserved.hacks' <=> 10.0.0.8 failed
00:19:21: DYNDNSUPD: Another update completed (total outstanding=0)
```

Configuring DHCP Support of DDNS Updates

DDNS updates contain information about A or forward RRs for a particular IP address. The IP address is in dotted decimal form, and there must be at least one A record for each host address. The name specified is the hostname expressed as an FQDN (ns.example.com). The PTR or reverse RRs map a domain name to another domain name and is used for reverse mapping (IP address to domain name).

The updates are performed using messages. In general, you will probably want DDNS updates done by the server after the server has sent the ACK response to the DHCP client. Performing the DDNS updates before sending the ACK response will delay the response to the client. Both methods are supported. The default is to do the updates after sending the response.

When looking for a client hostname to use in the update, the server will take the hostname from the FQDN option, if such exists, first. If there is no FQDN option, the server will look for a HOSTNAME option and take the name from there.

If the FQDN or HOSTNAME option is included in subsequent RENEWAL messages, the server will attempt to perform the DDNS update each time the lease is renewed. This process gives the opportunity for the client to change the name specified after the lease has been granted and have the server do the appropriate updates. Although the server has this capability, the DHCP client will continue to use the same hostname throughout the duration of a lease.

The IP address of the server to update is discovered by sending a DNS query for records associated with the hostname to update. If such a record exists, the hostname of the master DNS server is extracted from this information. If no such record exists, the record, which should be included in the response, is used as the authoritative record for the zone where the hostname exists. In either case, once the master DNS server hostname is found, another query for A RRs is sent in order to discover the IP address of this server. The resulting IP address is used for sending updates.

Perform this task to configure the DDNS updates.

Before You Begin

In order for DDNS updates to discover the DNS server, in cases in which the user did not configure the server, the **ip name-server** command should be configured. This name server should be reachable by the system, and the **ip domain lookup** command should be configured (which is the default anyway). In cases in which the configured hostname does not include a period (is not a fully qualified domain name [FQDN]), an IP domain name should be configured.

Note DHCP server-pool configuration commands and interface configurations have precedence over global configurations.

SUMMARY STEPS

1. **enable**
2. **configure terminal**
3. **ip dhcp update dns** [**both**] [**override**] [**before**]
4. **ip dhcp-client update dns** [**server** {**both** | **none**}]
5. **exit**

DETAILED STEPS

	Command or Action	Purpose
Step 1	**enable** **Example:** `Router> enable`	Enables privileged EXEC mode. • Enter your password if prompted.
Step 2	**configure terminal** **Example:** `Router# configure terminal`	Enters global configuration mode.
Step 3	**ip dhcp update dns** [**both**] [**override**] [**before**] **Example:** `Router(config)# ip dhcp update dns both override`	Enables DDNS updates of PTR RRs for all address pools except those configured with the per-pool **update dns** command, which overrides global configuration. The keywords are as follows: • **both** --(Optional) Enables the DHCP server to perform DDNS updates for A and PTR RRs, unless the DHCP client has specified in the FQDN option that the server should not perform the updates. • **override** --(Optional) Enables the DHCP server to perform DDNS updates for PTR RRs even if the DHCP client has specified in the FQDN option that the server should not perform the updates. **Note** If you specify the **both** and **override** keywords together, this enables the DHCP server to perform DDNS updates for A and PTR RRs overriding anything the DHCP client specified in the FQDN option to the contrary. • **before** --(Optional) Enables the DHCP server to perform DDNS updates before sending the DHCP ACK back to the client. The default is to perform updates after sending the DHCP ACK.
Step 4	**ip dhcp-client update dns** [**server** {**both**	**none**}] **Example:** `Router(config)# ip dhcp-client update dns server both`	Enables DDNS updates of PTR RRs. The optional **server** keyword enables the server to perform DDNS updates for A and PTR RRs. The keywords are as follows: • **both** --Enables the DHCP server to perform DDNS updates for A and PTR RRs, unless the DHCP client specifies in the FQDN option that the server should not perform the updates.

	Command or Action	Purpose
		• **none** --Enables the DHCP client to perform DDNS updates and the server will not perform any updates. The server can override this action. **Note** The **ip dhcp-client update dns server none** command instructs the server not to perform any updates. If configured to do so, the server can override the client. **Note** The **ip dhcp-client update dns server both** command instructs the server to update both the A and PTR RRs.
Step 5	**exit** **Example:** `Router(config)# exit`	Exits to privileged EXEC mode.

Examples

The following example shows how to configure A and PTR RR updates that are performed by the server only:

```
ip dhcp-client update dns server both
ip dhcp update dns both override
```

Configuring DDNS Update Support on Interfaces

Perform this task to configure your interfaces for DDNS update capability.

Note The interface configuration overrides the global configuration.

Before You Begin

In order for DDNS updates to discover the DNS server, in cases in which the user did not configure the server, the **ip name-server** command should be configured. This name server should be reachable by the system, and the **ip domain lookup** command should be configured (which is the default anyway). In cases in which the configured hostname does not include a period (is not a fully qualified domain name [FQDN]), an IP domain name should be configured.

Note The changes will not take effect until any current lease on the interface is released and a new lease is requested that uses a new DHCP DISCOVER packet. This means configuring the **ip address dhcp** command or using the **release dhcp** EXEC command followed by the **renew dhcp** EXEC command.

>

SUMMARY STEPS

1. **enable**
2. **configure terminal**
3. **interface** *interface-type number*
4. **ip dhcp client update dns** [**server** {**both** | **none**}]
5. **ip address dhcp**
6. **exit**

DETAILED STEPS

	Command or Action	Purpose
Step 1	**enable** **Example:** `Router> enable`	Enables privileged EXEC mode. • Enter your password if prompted.
Step 2	**configure terminal** **Example:** `Router# configure terminal`	Enters global configuration mode.
Step 3	**interface** *interface-type number* **Example:** `Router(config)# interface ethernet1`	Specifies an interface type and number and enters interface configuration mode.
Step 4	**ip dhcp client update dns** [**server** {**both** | **none**}] **Example:** `Router(config-if)# ip dhcp client update dns server both`	Configures the DHCP client to include an FQDN option when sending packets to the DHCP server. The keywords are as follows: • **both** --(Optional) Enables the DHCP server to perform DDNS updates for A and PTR RRs, unless the DHCP client specifies in the FQDN option that the server should not perform the updates. • **none** --(Optional) Enables the DHCP client to perform DDNS updates and the server will not perform any updates. The server can override this action. **Note** The **ip dhcp client update dns server none** command instructs the server not to perform any updates. If configured to do so, the server can override the client. **Note** The **ip dhcp client update dns server both** command instructs the server to update both the A and PTR RRs.
Step 5	**ip address dhcp**	Releases any current lease on the interface and enables the configuration.

	Command or Action	Purpose
	Example: `Router(config-if)# ip address dhcp`	**Note** You can also release any lease by using the **release dhcp** EXEC command followed by the **renew dhcp** EXEC command.
Step 6	**exit** **Example:** `Router(config-if)# exit`	Exits to privileged EXEC mode.

Configuring a Pool of DHCP Servers to Support DDNS Updates

There are two parts to the DDNS update configuration on the client side. First, if the **ip ddns update method** command is configured on the client, which specifies the DDNS-style updates, then the client will be trying to generate or perform A updates. If the **ip ddns update method ddns both** command is configured, then the client will be trying to update both A and PTR RRs.

Second, the only way for the client to communicate with the server, with reference to what updates it is generating or expecting the server to generate, is to include an FQDN option when communicating with the server. Whether or not this option is included is controlled on the client side by the **ip dhcp-client update dns** command in global configuration mode or the **ip dhcp client update dns** command in interface configuration mode.

If the FQDN option is included in the DHCP interaction, then the client may instruct the server to update “reverse” (the default), “both”, or “none.” Obviously, if the **ip ddns update method** command is configured with the **ddns** and **both**keywords, then the FQDN option configuration should reflect an IP DHCP client update DNS server none, but you have to configure the system correctly.

Finally, even if the client instructs the server to update both or update none, the server can override the client request and do whatever it was configured to do anyway. If there is an FQDN option in the DHCP interaction as above, then server can communicate to the client that it was overridden, in which case the client will not perform the updates because it knows that the server has done the updates. Even if the server is configured to perform the updates after sending the ACK (the default), it can still use the FQDN option to instruct the client what updates it will be performing and thus the client will not do the same types of updates.

If the server is configured with the **update dns** command with or without any keywords, and if the server does not see an FQDN option in the DHCP interaction, then it will assume that the client does not understand DDNS and will automatically act as though it were configured to update both A and PTR RRs on behalf of the client.

Perform this task to configure a pool of DHCP servers to support DDNS updates.

Before You Begin

In order for DDNS updates to discover the DNS server, in cases in which the user did not configure the server, the **ip name-server** command should be configured. This name server should be reachable by the system, and the **ip domain lookup** command should be configured (which is the default anyway). In cases in which

the configured hostname does not include a period (is not a fully qualified domain name [FQDN]), an IP domain name should be configured.

SUMMARY STEPS

1. **enable**
2. **configure terminal**
3. **ip dhcp pool** *pool-name*
4. **update dns** [**both** | **never**] [**override**] [**before**]
5. **exit**

DETAILED STEPS

	Command or Action	Purpose
Step 1	**enable** **Example:** `Router> enable`	Enables privileged EXEC mode. • Enter your password if prompted.
Step 2	**configure terminal** **Example:** `Router# configure terminal`	Enters global configuration mode.
Step 3	**ip dhcp pool** *pool-name* **Example:** `Router(config)# ip dhcp pool test`	Assigns a name to a DHCP pool and enters DHCP configuration mode.
Step 4	**update dns** [**both** | **never**] [**override**] [**before**] **Example:** `Router(dhcp-config)# update dns never`	Enables DDNS update capability for a pool of DHCP servers for any addresses assigned from this address pool. If the server is configured using this command with or without any of the other keywords, and if the server does not see an FQDN option in the DHCP interaction, then it will assume that the client does not understand DDNS and act as though it were configured to update both A and PTR records on behalf of the client. The keywords are as follows: • **both** --(Optional) Perform forward and reverse updates. If the **before** optional keyword is specified along with the **both** keyword, the server can perform DDNS updates before sending the ACK back to the client. If the **override** optional keyword is specified with the **both** keyword, the server can override the client and update forward and reverse RRs.

	Command or Action	Purpose
		If the **override** and **before** optional keywords are specified with the **both** keyword, the server can override the client (forward and reverse updates) and perform the updates before sending the ACK. • **never** --(Optional) Never perform updates for this pool. • **override** --(Optional) Override the client FQDN flags. If the **before** optional keyword is specified, the updates will be performed before sending the ACK. • **before** --(Optional) Perform updates before sending the ACK.
Step 5	**exit** **Example:** `Router(dhcp-config)# exit`	Exits to global configuration mode.

Examples

The following example shows how to configure a pool of DHCP servers to perform updates for A and PTR RRs before the ACK is sent:

```
ip dhcp pool test
 update dns both before
```

Configuring the Update Method and Interval

Perform this task to specify the update method and interval maximum.

Before You Begin

In order for DDNS updates to discover the DNS server, in cases in which the user did not configure the server, the **ip name-server** command should be configured. This name server should be reachable by the system, and the **ip domain lookup** command should be configured (which is the default anyway). In cases in which the configured hostname does not include a period (is not a fully qualified domain name [FQDN]), an IP domain name should be configured.

SUMMARY STEPS

1. **enable**
2. **configure terminal**
3. **ip ddns update method** *method-name*
4. **interval minimum** *days hours minutes seconds*
5. **interval maximum** *days hours minutes seconds*
6. **ddns** [**both**]
7. **internal**
8. **http**
9. **add** *url*
10. **remove** *url*
11. exit
12. **exit**
13. **interface** *interface-type number*
14. **ip ddns update hosthame** *hostname*
15. ip ddns update *name*
16. **exit**

DETAILED STEPS

	Command or Action	Purpose
Step 1	**enable** **Example:** `Router> enable`	Enables privileged EXEC mode. • Enter your password if pron
Step 2	**configure terminal** **Example:** `Router# configure terminal`	Enters global configuration mode
Step 3	**ip ddns update method** *method-name* **Example:** `Router(config)# ip ddns update method myupdate`	Specifies the update method nam
Step 4	**interval minimum** *days hours minutes seconds* **Example:** `Router(DDNS-update-method)# interval minimum 1 0 0 0`	Configures a minimum update int • *days* --Range is from 0 to 3 • *hours* --Range is from 0 to 2 • *minutes* --Range is from 0 t

	Command or Action	Purpose
		• *seconds* --Range is from 0 to 59
Step 5	**interval maximum** *days hours minutes seconds* **Example:** `Router(DDNS-update-method)# interval maximum 1 0 0 0`	Configures a maximum update interv • *days* --Range is from 0 to 365. • *hours* --Range is from 0 to 24. • *minutes* --Range is from 0 to 60 • *seconds* --Range is from 0 to 60
Step 6	**ddns** [**both**] **Example:** `Router(DDNS-update-method)# ddns`	Configures DDNS as the update meth updated. **Note** You can specify DDNS or H must disable it by using the configuration, see Steps 7,8,
Step 7	**internal** **Example:** `Router(DDNS-update-method)# internal`	Specifies that an internal cache will b
Step 8	**http** **Example:** `Router(DDNS-update-method)# http`	Configures HTTP as the update meth
Step 9	**add** *url* **Example:** `Router(DDNS-HTTP)# add http://test:test@members.dyndns.org/nic/update?system=dyndns&hostname=<h>&myip=<a>`	Configures a URL that should be invo an IP address. The following exampl information using DynDNS.org: • http://userid:password@members You have to enter the URL string abo DynDNS.org website. The special cha to update and the IP address with wh **Note** Before entering the question together on your keyboard. T the ? as a help query.
Step 10	**remove** *url* **Example:** `Router(DDNS-HTTP)# remove http://test:test@members.dyndns.org/nic/update?system=dyndns&hostname=<h>&myip=<a>`	Configures a URL that should be inv address. The URL takes the same for

	Command or Action	Purpose
Step 11	exit **Example:** `Router(DDNS-HTTP)# exit`	Exits to update-method configura
Step 12	**exit** **Example:** `Router(DDNS-update-method)# exit`	Exits to global configuration mod
Step 13	**interface** *interface-type number* **Example:** `Router(config)# interface ether1`	Enters interface configuration mo
Step 14	**ip ddns update hostname** *hostname* **Example:** `Router(config-if)# ip ddns update hostname abc.dyndns.org`	Specifies a host to be used for the IP address of the interface. The *ho* (for example, DynDNS.org).
Step 15	ip ddns update *name* **Example:** `Router(config-if) ip ddns update myupdate`	Specifies the name of the update address changes on this interface.
Step 16	**exit** **Example:** `Router(config)# exit`	Exits to privileged EXEC mode.

Examples

The following example shows how to configure the update method, the maximum interval of the updates (globally), and configure the hostname on the interface:

```
ip ddns update method mytest
ddns
 http
!Before entering the question mark (?) character in the add http CLI, press the control
(Ctrl) key and the v key together on your keyboard. This will allow you to enter the ?
without the software interpreting the ? as a help query.

 add http://test:test@members.dyndns.org/nic/update?system=dyndns&hostname=<h>&myip=<a>

 interval maximum 1 0 0 0
```

```
 exit
interface ether1

 ip ddns update hostname abc.dyndns.org

 ip ddns update mytest
```

Verifying DDNS Updates

Use the **debug ip ddns update** command to verify that DDNS updates are being performed. There are several sample configurations and the debug output that would display for that scenario.

Sample Configuration #1

The following scenario has a client configured for IETF DDNS updating of A DNS RRs during which a DHCP server is expected to update the PTR DNS RR. The DHCP client discovers the DNS server to update using an SOA RR lookup since the IP address to the server to update is not specified. The DHCP client is configured to include an FQDN DHCP option and notifies the DHCP server that it will be updating the A RRs.

```
!Configure the DHCP Client
ip ddns update method testing
 ddns
interface Ethernet1
 ip dhcp client update dns
 ip ddns update testing
 ip address dhcp
end
!Configure the DHCP Server
ip dhcp pool test
 network 10.0.0.0 255.0.0.0
 update dns
!Enable Debugging
Router# debug ip ddns update
00:14:39:%DHCP-6-ADDRESS_ASSIGN: Interface Ethernet1 assigned DHCP address 10.0.0.4, mask
255.0.0.0, hostname canada_reserved
00:14:39: DYNDNSUPD: Adding DNS mapping for canada_reserved.hacks <=> 10.0.0.4
00:14:39: DYNDNSUPD: Sleeping for 3 seconds waiting for interface Ethernet1 configuration
to settle
00:14:42: DHCPC: Server performed PTR update
00:14:42: DDNS: Enqueuing new DDNS update 'canada_reserved.hacks' <=> 10.0.0.4
00:14:42: DDNS: Zone name for 'canada_reserved.hacks' is 'hacks'
00:14:42: DDNS: Dynamic Update 1: (sending to server 10.19.192.32)
00:14:42: DDNS:   Zone = hacks
00:14:42: DDNS:   Prerequisite: canada_reserved.hacks not in use
00:14:42: DDNS:   Update: add canada_reserved.hacks IN A 10.0.0.4
00:14:42: DDNS: Dynamic DNS Update 1 (A) for host canada_reserved.hacks returned 0 (NOERROR)
00:14:42: DDNS: Update of 'canada_reserved.hacks' <=> 10.0.0.4 finished
00:14:42: DYNDNSUPD: Another update completed (total outstanding=0)
```

Sample Configuration #2

The following scenario has the client configured for IETF DDNS updating of both A and DNS RRs and requesting that the DHCP server update neither. The DHCP client discovers the DNS server to update using an SOA RR lookup since the IP address to the server to update is not specified. The DHCP client is configured to include an FQDN DHCP option that instructs the DHCP server not to update either A or PTR RRs. This is configured using the global version of the command.

```
!Configure the DHCP Client
ip dhcp-client update dns server none
ip ddns update method testing
 ddns both
interface Ethernet1
 ip ddns update testing
```

```
 ip address dhcp
end
!Configure the DHCP Server
ip dhcp pool test
 network 10.0.0.0 255.0.0.0
 update dns
!Enable Debugging
Router# debug ip ddns update
00:15:33:%DHCP-6-ADDRESS_ASSIGN: Interface Ethernet1 assigned DHCP address 10.0.0.5, mask
255.0.0.0, hostname canada_reserved
00:15:33: DYNDNSUPD: Adding DNS mapping for canada_reserved.hacks <=> 10.0.0.5
00:15:33: DYNDNSUPD: Sleeping for 3 seconds waiting for interface Ethernet1 configuration
to settle
00:15:36: DDNS: Enqueuing new DDNS update 'canada_reserved.hacks' <=> 10.0.0.5
00:15:36: DDNS: Zone name for '10.0.0.11.in-addr.arpa.' is '10.in-addr.arpa'
00:15:36: DDNS: Dynamic Update 1: (sending to server 10.19.192.32)
00:15:36: DDNS:   Zone = 10.in-addr.arpa
00:15:36: DDNS:   Prerequisite: 10.0.0.11.in-addr.arpa. not in use
00:15:36: DDNS:   Update: add 10.0.0.11.in-addr.arpa. IN PTR canada_reserved.hacks
00:15:36: DDNS: Dynamic DNS Update 1 (PTR) for host canada_reserved.hacks returned 0 (NOERROR)
00:15:36: DDNS: Zone name for 'canada_reserved.hacks' is 'hacks'
00:15:36: DDNS: Dynamic Update 1: (sending to server 10.19.192.32)
00:15:36: DDNS:   Zone = hacks
00:15:36: DDNS:   Prerequisite: canada_reserved.hacks not in use
00:15:36: DDNS:   Update: add canada_reserved.hacks IN A 10.0.0.5
00:15:36: DDNS: Dynamic DNS Update 1 (A) for host canada_reserved.hacks returned 0 (NOERROR)
00:15:36: DDNS: Update of 'canada_reserved.hacks' <=> 10.0.0.5 finished
00:15:36: DYNDNSUPD: Another update completed (total outstanding=0)
```

Sample Configuration #3

The following scenario the client is configured for IETF DDNS updating of both A and DNS RRs and requesting that the DHCP server update neither. The DHCP client explicitly specifies the server to update. The DHCP client is configured to include an FQDN DHCP option which instructs the DHCP server not to update either A or PTR RRs. This is configured using the global version of the command. The DHCP server is configured to override the client request and update both A and PTR RR anyway.

```
!Configure the DHCP Client
ip dhcp client update dns server non
ip ddns update method testing
 ddns both
interface Ethernet1
 ip dhcp client update dns server none
 ip ddns update testing
 ip address dhcp
end
!Configure the DHCP Server
ip dhcp pool test
 network 10.0.0.0 255.0.0.0
 update dns both override
!Enable Debugging on the DHCP Client
Router# debug ip ddns update
00:16:30:%DHCP-6-ADDRESS_ASSIGN: Interface Ethernet1 assigned DHCP address 10.0.0.6, mask
255.0.0.0, hostname canada_reserved
00:16:30: DYNDNSUPD: Adding DNS mapping for canada_reserved.hacks <=> 10.0.0.6
00:16:30: DYNDNSUPD: Sleeping for 3 seconds waiting for interface Ethernet1 configuration
to settle
00:16:33: DHCPC: Server performed both updates
```

Sample Configuration #4

In the following scenario the client is configured for IETF DDNS updating of both A and DNS RRs and requesting the DHCP server to update neither. The DHCP client explicitly specifies the server to update. The DHCP client is configured to include an FQDN DHCP option which instructs the DHCP server not to update

either A or PTR RRs. This is configured using the global version of the command. The DHCP server is configured to allow the client to update whatever RR it chooses.

```
!Configure the DHCP Client
ip dhcp client update dns server non
ip ddns update method testing
 ddns both
interface Ethernet1
 ip dhcp client update dns server none
 ip ddns update testing host 172.19.192.32
 ip address dhcp
end
!Configure the DHCP Server
ip dhcp pool test
 network 10.0.0.0 255.0.0.0
 update dns
!Enable Debugging on the DHCP Client
Router# debug ip ddns update
00:17:52:%DHCP-6-ADDRESS_ASSIGN: Interface Ethernet1 assigned DHCP address 10.0.0.7, mask
255.0.0.0, hostname canada_reserved
00:17:52: DYNDNSUPD: Adding DNS mapping for canada_reserved.hacks <=> 10.0.0.6
00:17:52: DYNDNSUPD: Sleeping for 3 seconds waiting for interface Ethernet1 configuration
to settle
00:17:55: DDNS: Enqueuing new DDNS update 'canada_reserved.hacks' <=> 10.0.0.7
00:17:55: DYNDNSUPD: Adding DNS mapping for canada_reserved.hacks <=> 10.0.0.7 server
10.19.192.32
00:17:55: DDNS: Enqueuing new DDNS update 'canada_reserved.hacks' <=> 10.0.0.7 server
10.19.192.32
00:17:55: DDNS: Zone name for '10.0.0.11.in-addr.arpa.' is '11.in-addr.arpa'
00:17:55: DDNS: Dynamic Update 1: (sending to server 10.19.192.32)
00:17:55: DDNS: Zone = 10.in-addr.arpa
00:17:55: DDNS: Prerequisite: 10.0.0.11.in-addr.arpa. not in use
00:17:55: DDNS: Update: add 10.0.0.11.in-addr.arpa. IN PTR canada_reserved.hacks
00:17:55: DDNS: Zone name for '10.0.0.11.in-addr.arpa.' is '10.in-addr.arpa'
00:17:55: DDNS: Using server 10.19.192.32
00:17:55: DDNS: Dynamic Update 1: (sending to server 10.19.192.32)
00:17:55: DDNS: Zone = 10.in-addr.arpa
00:17:55: DDNS: Prerequisite: 10.0.0.11.in-addr.arpa. not in use
00:17:55: DDNS: Update: add 10.0.0.11.in-addr.arpa. IN PTR canada_reserved.hacks
00:17:55: DDNS: Dynamic DNS Update 1 (PTR) for host canada_reserved.hacks returned 0 (NOERROR)
00:17:55: DDNS: Dynamic DNS Update 1 (PTR) for host canada_reserved.hacks returned 6
(YXDOMAIN)
00:17:55: DDNS: Dynamic Update 2: (sending to server 10.19.192.32)
00:17:55: DDNS: Zone = 10.in-addr.arpa
00:17:55: DDNS: Update: delete 10.0.0.11.in-addr.arpa. all PTR RRs
00:17:55: DDNS: Update: add 10.0.0.11.in-addr.arpa. IN PTR canada_reserved.hacks
00:17:55: DDNS: Dynamic DNS Update 2 (PTR) for host canada_reserved.hacks returned 0 (NOERROR)
00:17:55: DDNS: Zone name for 'canada_reserved.hacks' is 'hacks'
00:17:55: DDNS: Dynamic Update 1: (sending to server 10.19.192.32)
00:17:55: DDNS: Zone = hacks
00:17:55: DDNS: Prerequisite: canada_reserved.hacks not in use
00:17:55: DDNS: Update: add canada_reserved.hacks IN A 10.0.0.7
00:17:55: DDNS: Dynamic DNS Update 1 (A) for host canada_reserved.hacks returned 0 (NOERROR)
00:17:55: DDNS: Update of 'canada_reserved.hacks' <=> 10.0.0.7 finished
00:17:55: DYNDNSUPD: Another update completed (total outstanding=1)
00:17:55: DDNS: Zone name for 'canada_reserved.hacks' is 'hacks'
00:17:55: DDNS: Using server 10.19.192.32
00:17:55: DDNS: Dynamic Update 1: (sending to server 10.19.192.32)
00:17:55: DDNS: Zone = hacks
00:17:55: DDNS: Prerequisite: canada_reserved.hacks not in use
00:17:55: DDNS: Update: add canada_reserved.hacks IN A 10.0.0.7
00:17:55: DDNS: Dynamic DNS Update 1 (A) for host canada_reserved.hacks returned 6 (YXDOMAIN)
00:17:55: DDNS: Dynamic Update 2: (sending to server 10.19.192.32)
00:17:55: DDNS: Zone = hacks
00:17:55: DDNS: Update: delete canada_reserved.hacks all A RRs
00:17:55: DDNS: Update: add canada_reserved.hacks IN A 10.0.0.7
00:17:55: DDNS: Dynamic DNS Update 2 (A) for host canada_reserved.hacks returned 0 (NOERROR)
00:17:55: DDNS: Update of 'canada_reserved.hacks' <=> 10.0.0.7 finished
00:17:55: DYNDNSUPD: Another update completed (total outstanding=0)
```

Sample Configuration #5

In the following scenario, the debug output is displaying internal host table updates when the default domain name is “hacks.” The “test” update method specifies that the internal Cisco IOS host table should be updated. Configuring the update method as “test” should be used when the address on the Ethernet 0/0 interface changes. The hostname is configured for the update on this interface.

```
ip domain name hacks
ip ddns update method test
 internal
interface ethernet0/0
 ip ddns update test hostname test2
 ip addr dhcp
!Enable Debugging
Router# debug ip ddns update
*Jun 4 03:11:10.591:%DHCP-6-ADDRESS_ASSIGN: Interface Ethernet0/0 assigned DHCP address
10.0.0.5, mask 255.0.0.0, hostname test2
*Jun 4 03:11:10.591: DYNDNSUPD: Adding DNS mapping for test2.hacks <=> 10.0.0.5
*Jun 4 03:11:10.591: DYNDNSUPD: Adding internal mapping test2.hacks <=> 10.0.0.5
```

Using the **show hosts** command displays the newly added host table entry.

```
Router# show hosts
Default domain is hacks
Name/address lookup uses domain service
Name servers are 255.255.255.255
Codes: UN - unknown, EX - expired, OK - OK,?? - revalidate
       temp - temporary, perm - permanent
       NA - Not Applicable None - Not defined
Host                      Port Flags      Age Type   Address(es)
test2.hacks               None (perm, OK) 0   IP    10.0.0.5
```

Shutting down the interface removes the host table entry.

```
interface ethernet0/0
 shutdown
*Jun 4 03:14:02.107: DYNDNSUPD: Removing DNS mapping for test2.hacks <=> 10.0.0.5
*Jun 4 03:14:02.107: DYNDNSUPD: Removing mapping test2.hacks <=> 10.0.0.5
```

The **show hosts** command output shows the entry has been removed.

```
Router# show hosts
Default domain is hacks
Name/address lookup uses domain service
Name servers are 255.255.255.255
Codes: UN - unknown, EX - expired, OK - OK,?? - revalidate
       temp - temporary, perm - permanent
       NA - Not Applicable None - Not defined
Host                      Port Flags      Age Type   Address(es)
```

Sample Configuration #6

In the following scenario, the debug output shows the HTTP-style DDNS updates. The sample configuration defines a new IP DDNS update method named dyndns that configures a URL to use when adding or changing an address. No URL has been defined for use when removing an address since DynDNS.org does not use such a URL for free accounts. A maximum update interval of 28 days has been configured, so specifying that updates should be sent at least every 28 days. Configuring the new dyndns update method should be used for Ethernet interface .

Note Before entering the question mark (?) character in the “add http” configuration after the **update** keyword, press the control (Ctrl) key and the “v” key together on your keyboard. This will allow you to enter the ? without the software interpreting it as a help query.

```
!Configure the DHCP Client
ip ddns update method dyndns
 http
    add http://test:test@<s>/nic/update?system=dyndns&hostname=<h>&myip=<a>
    interval max 28 0 0 0
interface ethernet1
 ip ddns update hostname test.dyndns.org
 ip ddns update dyndns host members.dyndns.org
 ip addr dhcp
!Enable Debugging
Router# debug ip ddns update
00:04:35:%DHCP-6-ADDRESS_ASSIGN: Interface Ethernet1 assigned DHCP address 10.32.254.187,
mask 255.255.255.240, hostname test.dyndns.org
00:04:35: DYNDNSUPD: Adding DNS mapping for test.dyndns.org <=> 10.32.254.187 server
10.208.196.94
00:04:35: DYNDNSUPD: Sleeping for 3 seconds waiting for interface Ethernet1 configuration
to settle
00:04:38: HTTPDNS: Update add called for test.dyndns.org <=> 10.32.254.187
00:04:38: HTTPDNS: Update called for test.dyndns.org <=> 10.32.254.187
00:04:38: HTTPDNS: init
00:04:38: HTTPDNSUPD: Session ID = 0x7
00:04:38: HTTPDNSUPD: URL =
'http://test:test@10.208.196.94/nic/update?system=dyndns&hostname=test.dyndns.org&myip=10.32.254.187'
00:04:38: HTTPDNSUPD: Sending request
00:04:40: HTTPDNSUPD: Response for update test.dyndns.org <=> 10.32.254.187
00:04:40: HTTPDNSUPD: DATA START
good 10.32.254.187
00:04:40: HTTPDNSUPD: DATA END, Status is Response data received, successfully
00:04:40: HTTPDNSUPD: Call returned SUCCESS for update test.dyndns.org <=> 10.32.254.187
00:04:40: HTTPDNSUPD: Freeing response
00:04:40: DYNDNSUPD: Another update completed (outstanding=0, total=0)
00:04:40: HTTPDNSUPD: Clearing all session 7 info
!28 days later, the automatic update happens.
00:05:39: DYNDNSUPD: Adding DNS mapping for test.dyndns.org <=> 10.32.254.187 server
10.208.196.94
00:05:39: HTTPDNS: Update add called for test.dyndns.org <=> 10.32.254.187
00:05:39: HTTPDNS: Update called for test.dyndns.org <=> 10.32.254.187
00:05:39: HTTPDNS: init
00:05:39: HTTPDNSUPD: Session ID = 0x8
00:05:39: HTTPDNSUPD: URL =
'http://test:test@10.208.196.94/nic/update?system=dyndns&hostname=test.dyndns.org&myip=10.32.254.187'
00:05:39: HTTPDNSUPD: Sending request
00:05:39: HTTPDNSUPD: Response for update test.dyndns.org <=> 10.32.254.187
00:05:39: HTTPDNSUPD: DATA START
nochg 10.32.254.187
00:05:39: HTTPDNSUPD: DATA END, Status is Response data received, successfully
00:05:39: HTTPDNSUPD: Call returned SUCCESS for update test.dyndns.org <=> 10.32.254.187
00:05:39: HTTPDNSUPD: Freeing response
00:05:39: DYNDNSUPD: Another update completed (outstanding=0, total=0)
00:05:39: HTTPDNSUPD: Clearing all session 8 info
```

Configuration Examples for Dynamic DNS Support for Cisco IOS Software

Configuration of the DHCP Client Example

The following example shows that no DDNS updates will be performed for addresses assigned from the address pool “abc.” Addresses allocated from the address pool “def” will have both forward (A) and reverse (PTR) updates performed. This configuration has precedence over the global server configurations.

```
ip dhcp update dns both override
 ip dhcp pool abc
  network 10.1.0.0 255.255.0.0
!
 update dns never
!
 ip dhcp pool def
  network 10.10.0.0 255.255.0.0
```

Configuration of the DHCP Server Example

The following example shows how to configure A and PTR RR updates that are performed by the server only:

```
ip dhcp-client update dns server both

ip dhcp update dns both override
```

Configuration of the HTTP Updates Example

The following example shows how to configure a PPPoE server for HTTP DDNS:

```
!Username and Password for PPP Authentication Configuration
!
username user1 password 0 cisco
!
!DHCP Pool Configuration
ip dhcp pool mypool
 network 10.10.10.0 255.255.255.0
 default-router 10.10.10.1
!
!VPDN configuration for PPPoE
vpdn enable
!
vpdn-group pppoe
accept-dialin
protocol pppoe
virtual-template 1
!
interface Loopback0
ip address 10.10.10.1 255.255.255.0
!
!Port used to connect to the Internet, it can be the same port that is under test, but to
make the test clear and simple these two are separated.
!
interface FastEthernet0/0
 ip address 10.0.58.71 255.255.255.0
```

```
!
!Port under test.
!
interface FastEthernet0/1
 no ip address
 pppoe enable
!
!Virtual template and address pool config for PPPoE.
interface Virtual-Template1
 ip unnumbered Loopback0
 ip mtu 1492
 peer default ip address dhcp-pool mypool
 ppp authentication chap
```

The following example shows how to configure a DHCP client for IETF DDNS:

```
!Default hostname of the router.
hostname mytest
!
!Default domain name on the router.
ip domain name test.com
!
!Port under test.
!
interface FastEthernet0/1
 no ip address (configured to “ip address dhcp”)
```

The following example shows how to configure the method of update and the maximum interval of the updates (globally) and configure the hostname on the interface:

Note Before entering the question mark (?) character in the “add http” configuration after the **update** keyword, press the control (Ctrl) key and the “v” key together on your keyboard. This will allow you to enter the ? without the software interpreting it as a help query.

```
ip ddns update method mytest
ddns
 http

 add http://test:test@members.dyndns.org/nic/update?system=dyndns&hostname=<h>&myip=<a>

 interval maximum 1 0 0 0
 exit
interface ether1

 ip ddns update hostname abc.dyndns.org

 ip ddns update mytest
```

The following are examples of URLs that can be used to update some HTTP DNS update services. These URLs are correct to the best of the knowledge of Cisco but have not been tested in all cases. Where the word “USERNAME:” appears in the URL, the customer account username at the HTTP site should be used.

Where the word “PASSWORD” appears in the URL, the customer password for that account should be used:

Note Before entering the question mark (?) character in the “add http” configuration after the **update** keyword, press the control (Ctrl) key and the “v” key together on your keyboard. This will allow you to enter the ? without the software interpreting it as a help query.

DDNS

```
http://USERNAME:PASSWORD@members.dyndns.org/nic/update?system=dyndns&hostname=<h>&myip=<a>
!Requires “interval max 28 0 0 0" in the update method definition.
```

TZO

```
http://cgi.tzo.com/webclient/signedon.html?TZOName=<h>&Email=USERNAME&TZOKey=PASSWORD&IPAddress=<a>
```

EASYDNS

```
http://USERNAME:PASSWORD@members.easydns.com/dyn/ez-ipupdate.php?action=edit&myip=<a>&host_id=<h>
```

JUSTLINUX

```
http://USERNAME:PASSWORD@www.justlinux.com/bin/controlpanel/dyndns/jlc.pl?direst=1&username=USERNAME&password=PASSWORD&host=<h>&ip=<a>
```

DYNS

```
http://USERNAME:PASSWORD@www.dyns.cx/postscript.php?username=USERNAME&password=PASSWORD&host=<h>&ip=<a>
```

HN

```
http://USERNAME:PASSWORD@dup.hn.org/vanity/update?ver=1&IP=<a>
```

ZONEEDIT

```
http://USERNAME:PASSWORD@www.zoneedit.com/auth/dynamic.html?host=<h>&dnsto=<a>
```

Note Because these services are provided by the respective companies, the URLs may be subject to change or the service could be discontinued at any time. Cisco takes no responsibility for the accuracy or use of any of this information. The URLs were obtained using an application called “ez-ipupdate,” which is available for free on the Internet.

Additional References

The following sections provide references related to the Dynamic DNS Support for Cisco IOS Software feature.

Related Documents

Related Topic	Document Title
DNS Configuration Tasks	“Configuring DNS” module
DNS commands: complete command syntax, command mode, command history, defaults, usage guidelines, and examples	*Cisco IOS IP Addressing Services Command Reference*

Standards

Standards	Title
No new or modified standards are supported by this feature, and support for existing standards has not been modified by this feature.	--

MIBs

MIBs	MIBs Link
No new or modified MIBs are supported by this feature, and support for existing MIBs has not been modified by this feature.	To locate and download MIBs for selected platforms, Cisco IOS releases, and feature sets, use Cisco MIB Locator found at the following URL: http://www.cisco.com/go/mibs

RFCs

RFCs	Title
RFC 2136	*Dynamic Updates in the Domain Name System (DNS Update)*
RFC 3007	*Secure Domain Name System (DNS) Dynamic Update*

Technical Assistance

Description	Link
The Cisco Support and Documentation website provides online resources to download documentation, software, and tools. Use these resources to install and configure the software and to troubleshoot and resolve technical issues with Cisco products and technologies. Access to most tools on the Cisco Support and Documentation website requires a Cisco.com user ID and password.	http://www.cisco.com/cisco/web/support/index.html

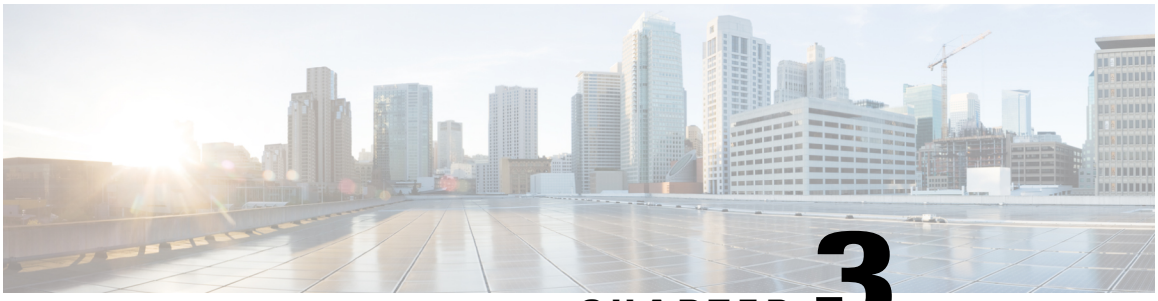
Feature Information for Dynamic DNS Support for Cisco IOS Software

The following table provides release information about the feature or features described in this module. This table lists only the software release that introduced support for a given feature in a given software release train. Unless noted otherwise, subsequent releases of that software release train also support that feature.

Use Cisco Feature Navigator to find information about platform support and Cisco software image support. To access Cisco Feature Navigator, go to www.cisco.com/go/cfn. An account on Cisco.com is not required.

Table 2: Feature Information for Dynamic DNS Support for Cisco IOS Software

Feature Name	Releases	Feature Information
Dynamic DNS Support for Cisco IOS Software	12.3(8)YA 12.3(14)T	The Dynamic DNS Support for Cisco IOS Software feature enables Cisco IOS software devices to perform Dynamic Domain Name System (DDNS) updates to ensure that an IP host DNS name is correctly associated with its IP address.



CHAPTER 3

VRF-Aware DNS

The VRF-Aware DNS feature enables the configuration of a Virtual Private Network (VPN) routing and forwarding instance (VRF) table so that the domain name system (DNS) can forward queries to name servers using the VRF table rather than the named DNS server in the global IP address space. This feature allows DNS requests to be resolved within the appropriate Multiprotocol Label Switching (MPLS) VPN.

Note You can specify IPv4 and IPv6 addresses while performing various tasks in this feature. The resource record type AAAA is used to map a domain name to an IPv6 address. The IP6.ARPA domain is defined to look up a record given an IPv6 address.

- Finding Feature Information, page 45
- Information About VRF-Aware DNS, page 46
- How to Configure VRF-Aware DNS, page 47
- Configuration Examples for VRF-Aware DNS, page 51
- Additional References, page 52
- Feature Information for VRF-Aware DNS, page 53

Finding Feature Information

Your software release may not support all the features documented in this module. For the latest caveats and feature information, see Bug Search Tool and the release notes for your platform and software release. To find information about the features documented in this module, and to see a list of the releases in which each feature is supported, see the feature information table at the end of this module.

Use Cisco Feature Navigator to find information about platform support and Cisco software image support. To access Cisco Feature Navigator, go to www.cisco.com/go/cfn. An account on Cisco.com is not required.

Information About VRF-Aware DNS

Domain Name System

Domain Name System (DNS) is a standard that defines a domain naming procedure used in TCP/IP. A domain is a hierarchical separation of the network into groups and subgroups with domain names identifying the structure. The named groups consist of named objects, usually devices like IP hosts, and the subgroups are domains. DNS has three basic functions:

- Name space: This function is a hierarchical space organized from a single root into domains. Each domain can contain device names or more specific information. A special syntax defines valid names and identifies the domain names.
- Name registration: This function is used to enter names into the DNS database. Policies are outlined to resolve conflicts and other issues.
- Name resolution: This function is a distributed client and server name resolution standard. The name servers are software applications that run on a server and contain the resource records (RRs) that describe the names and addresses of those entities in the DNS name space. A name resolver is the interface between the client and the server. The name resolver requests information from the server about a name. A cache can be used by the name resolver to store learned names and addresses.

A DNS server can be a dedicated device or a software process running on a device. The server stores and manages data about domains and responds to requests for name conflict resolutions. In a large DNS implementation, there can be a distributed database over many devices. A server can be a dedicated cache.

VRF Mapping and VRF-Aware DNS

To keep track of domain names, IP has defined the concept of a name server, whose job is to hold a cache (or database) of names appended to IP addresses. The cached information is important because the requesting DNS will not need to query for that information again, which is why DNS works well. If a server had to query each time for the same address because it had not saved any data, the queried servers would be flooded and would crash.

A gateway for multiple enterprise customers can be secured by mapping the remote users to a VRF domain. Mapping means obtaining the IP address of the VRF domain for the remote users. By using VRF domain mapping, a remote user can be authenticated by a VRF domain-specific AAA server so that the remote-access traffic can be forwarded within the VRF domain to the servers on the corporate network.

To support traffic for multiple VRF domains, the DNS and the servers used to resolve conflicts must be VRF aware. VRF aware means that a DNS subsystem will query the VRF name cache first, then the VRF domain, and store the returned RRs in a specific VRF name cache. Users are able to configure separate DNS name servers per VRF.

VRF-aware DNS forwards queries to name servers using the VRF table. Because the same IP address can be associated with different DNS servers in different VRF domains, a separate list of name caches for each VRF is maintained. The DNS looks up the specific VRF name cache first, if a table has been specified, before sending a query to the VRF name server. All IP addresses obtained from a VRF-specific name cache are routed using the VRF table.

How to Configure VRF-Aware DNS

Defining a VRF Table and Assigning a Name Server to Enable VRF-Aware DNS

Perform this task to define a VRF table and assign a name server.

A VRF-specific name cache is dynamically created if one does not exist whenever a VRF-specific name server is configured by using the **ip name-server vrf**command option or a permanent name entry is configured by using the **ip host vrf**command option. The VRF name cache is removed whenever all name server and permanent entries in the VRF are disabled.

It is possible that multiple name servers are configured with the same VRF name. The system will send queries to those servers in turn until any of them responds, starting with the server that sent a response the last time.

SUMMARY STEPS

1. **enable**
2. **configure terminal**
3. **ip vrf** *vrf-name*
4. **rd** *route-distinguisher*
5. **exit**
6. **ip name-server** [**vrf** *vrf-name*] *server-address1* [*server-address2...server-address6*]
7. **ip domain lookup** [**source-interface** *interface-type interface-number*]

DETAILED STEPS

	Command or Action	Purpose
Step 1	**enable** **Example:** `Device> enable`	Enables privileged EXEC mode. • Enter your password if prompted.
Step 2	**configure terminal** **Example:** `Device# configure terminal`	Enters global configuration mode.
Step 3	**ip vrf** *vrf-name* **Example:** `Device(config)# ip vrf vpn1`	Defines a VRF table and enters VRF configuration mode. • The *vrf-name* argument can be up to 32 characters.

	Command or Action	Purpose
Step 4	**rd** *route-distinguisher* **Example:** `Device(config)# rd 100:21`	Creates routing and forwarding tables for a VRF.
Step 5	**exit** **Example:** `Device(config-vrf)# exit`	Exits VRF configuration mode.
Step 6	**ip name-server** [**vrf** *vrf-name*] *server-address1* [*server-address2...server-address6*] **Example:** `Device(config)# ip name-server vrf vpn1 172.16.1.111 2001:DB8:1::1`	Assigns the address of one or more name servers to a VRF table to use for name and address resolution. • The name server IP address can be an IPv4 or IPv6 address. • The **vrf** keyword is optional but must be specified if the name server is used with VRF. The *vrf-name* argument assigns a name to the VRF.
Step 7	**ip domain lookup** [**source-interface** *interface-type interface-number*] **Example:** `Device(config)# ip domain lookup`	(Optional) Enables DNS-based address translation. • DNS is enabled by default. You only need to use this command if DNS has been disabled.

Mapping VRF-Specific Hostnames to IP Addresses

Perform this task to map VRF-specific hostnames to IP addresses.

SUMMARY STEPS

1. **enable**
2. **configure terminal**
3. Do one of the following:
   - **ip domain name** [**vrf** *vrf-name*] *name*
   - **ip domain list** [**vrf** *vrf-name*] *name*

DETAILED STEPS

	Command or Action	Purpose
Step 1	**enable** **Example:** `Device> enable`	Enables privileged EXEC mode. • Enter your password if prompted.
Step 2	**configure terminal** **Example:** `Device# configure terminal`	Enters global configuration mode.

Step 3	Do one of the following:
	• **ip domain name** [**vrf** *vrf-name*] *name*
	• **ip domain list** [**vrf** *vrf-name*] *name*
	
	Example:
	`Device(config)# ip domain name vrf vpn1 cisco.com`
	
	Example:
	`Device(config)# ip domain list vrf vpn1 cisco.com`
		Defines a default domain name that the software will use to complete unqualified hostnames.
		or
		Defines a list of default domain names to complete unqualified hostnames.
		• You can specify a default domain name that the software will use to complete domain name requests. You can specify either a single domain name or a list of domain names. Any hostname that does not contain a complete domain name will have the default domain name you specify appended to it before the name is looked up.
		• The **vrf** keyword and *vrf-name* argument specify a default VRF domain name.
		• The **ip domain list** command can be entered multiple times to specify more than one domain name to append when doing a DNS query. The system will append each in turn until it finds a match.

Configuring a Static Entry in a VRF-Specific Name Cache

Perform this task to configure a static entry in a VRF-specific name cache.

A VRF-specific name cache is dynamically created if one does not exist whenever a name server is configured for the VRF by using the **ip name-server vrf**command option or a permanent name entry is configured by using the **ip host vrf** command option. The VRF name cache is removed whenever all name server and permanent entries in the VRF are disabled.

SUMMARY STEPS

1. **enable**
2. **configure terminal**
3. **ip host** [**vrf** *vrf-name*] *name* [*tcp-port*] *address1* [*address2...address8*

DETAILED STEPS

	Command or Action	Purpose
Step 1	**enable** **Example:** `Device> enable`	Enables privileged EXEC mode. • Enter your password if prompted.
Step 2	**configure terminal** **Example:** `Device# configure terminal`	Enters global configuration mode.
Step 3	**ip host** [**vrf** *vrf-name*] *name* [*tcp-port*] *address1* [*address2...address8* **Example:** **Example:** `Device(config)# ip host vrf vpn3 company1.com 172.16.2.1`	Defines a static hostname-to-address mapping in the host cache. • The IP address of the host can be an IPv4 or IPv6 address, and the IP address can be associated with a Virtual Private Network (VPN) routing and forwarding (VRF) instance. • If the **vrf** keyword and *vrf-name* arguments are specified, then a permanent entry is created only in the VRF-specific name cache.

Verifying the Name Cache Entries in the VRF Table

Perform this task to verify the name cache entries in the VRF table.

SUMMARY STEPS

1. **enable**
2. **show hosts** [**vrf** *vrf-name*] {**all**| *hostname*} [**summary**]
3. **clear host** [**vrf** *vrf-name*] {**all**| *hostname*}

DETAILED STEPS

	Command or Action	Purpose
Step 1	**enable** **Example:** `Device> enable`	Enables privileged EXEC mode. • Enter your password if prompted.

	Command or Action	Purpose
Step 2	**show hosts** [**vrf** *vrf-name*] {**all**| *hostname*} [**summary**] **Example:** `Device# show hosts vrf vpn2`	• Displays the default domain name, the style of name lookup service, a list of name server hosts, the cached list of hostnames and addresses, and the cached list of hostnames and addresses specific to a particular Virtual Private Network (VPN). • The **vrf** keyword and *vrf-name* argument only display the entries if a VRF name has been configured. • If you enter the **show hosts** command without specifying any VRF, only the entries in the global name cache will display.
Step 3	**clear host** [**vrf** *vrf-name*] {**all**| *hostname*} **Example:** `Device# clear host vrf vpn2`	(Optional) Deletes entries from the hostname-to-address global address cache or VRF name cache.

Configuration Examples for VRF-Aware DNS

Example: VRF-Specific Name Server Configuration

The following example shows how to specify a VPN named vpn1 with the IP addresses of 172.16.1.111 and 172.16.1.2 as the name servers:

```
ip name-server vrf vpn1 172.16.1.111 172.16.1.2
```

Example: VRF-Specific Domain Name List Configuration

The following example shows how to add several domain names to a list in vpn1 and vpn2. The domain name is only used for name queries in the specified VRF.

```
ip domain list vrf vpn1 company.com
ip domain list vrf vpn2 school.edu
```

If there is no domain list, the domain name that you specified with the **ip domain name** global configuration command is used. If there is a domain list, the default domain name is not used. The **ip domain list** command is similar to the **ip domain name** command, except that with the **ip domain list**command you can define a list of domains, each to be tried in turn until a match is found.

Example: VRF-Specific Domain Name Configuration

The following example shows how to define cisco.com as the default domain name for a VPN named vpn1. The domain name is only used for name queries in the specified VRF.

```
ip domain name vrf vpn1 cisco.com
```

Any IP hostname that does not contain a domain name (that is, any name without a dot) will have the dot and cisco.com appended to it before being looked up.

Example: VRF-Specific IP Host Configuration

The following example shows how to define two static hostname-to-address mappings in the host cache for vpn2 and vpn3:

```
ip host vrf vpn2 host2 10.168.7.18
ip host vrf vpn3 host3 10.12.0.2
```

Additional References

Related Documents

Related Topic	Document Title
DNS configuration tasks	"Configuring DNS" module
IP addressing services commands: complete command syntax, command mode, command history, defaults, usage guidelines, and examples	*Cisco IOS IP Addressing Services Command Reference*

Standards

Standards	Title
No new or modified standards are supported by this feature, and support for existing standards has not been modified by this feature.	--

MIBs

MIBs	MIBs Link
No new or modified MIBs are supported by this feature, and support for existing MIBs has not been modified by this feature.	To locate and download MIBs for selected platforms, Cisco IOS releases, and feature sets, use Cisco MIB Locator found at the following URL: http://www.cisco.com/go/mibs

RFCs

RFCs	Title
No new or modified RFCs are supported by this feature, and support for existing RFCs has not been modified by this feature.	--

Technical Assistance

Description	Link
The Cisco Support and Documentation website provides online resources to download documentation, software, and tools. Use these resources to install and configure the software and to troubleshoot and resolve technical issues with Cisco products and technologies. Access to most tools on the Cisco Support and Documentation website requires a Cisco.com user ID and password.	http://www.cisco.com/cisco/web/support/index.html

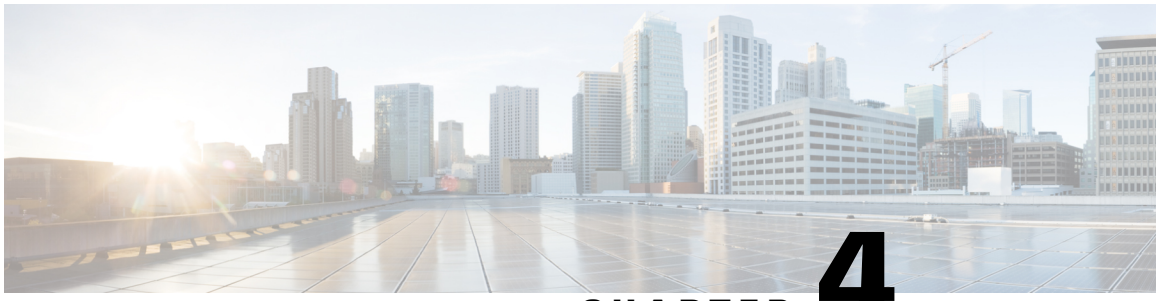
Feature Information for VRF-Aware DNS

The following table provides release information about the feature or features described in this module. This table lists only the software release that introduced support for a given feature in a given software release train. Unless noted otherwise, subsequent releases of that software release train also support that feature.

Use Cisco Feature Navigator to find information about platform support and Cisco software image support. To access Cisco Feature Navigator, go to www.cisco.com/go/cfn. An account on Cisco.com is not required.

Table 3: Feature Information for VRF-Aware DNS

Feature Name	Releases	Feature Information
VRF-Aware DNS	15.2(1)SY	The VRF-Aware DNS feature enables the configuration of a Virtual Private Network (VPN) routing and forwarding instance (VRF) table so that the domain name system (DNS) can forward queries to name servers using the VRF table rather than the named DNS server in the global IP address space. This feature allows DNS requests to be resolved within the appropriate Multiprotocol Label Switching (MPLS) VPN.



CHAPTER 4

Split DNS

The Split DNS feature enables a Cisco device to respond to Domain Name System (DNS) queries using a specific configuration and associated host table cache that are selected based on certain characteristics of the queries. In a Split DNS environment, multiple DNS databases can be configured on the device, and the Cisco IOS software can be configured to choose one of these DNS name server configurations whenever the device must respond to a DNS query by forwarding or resolving the query.

Note You can specify IPv4 and IPv6 addresses while performing various tasks in this feature. The resource record type AAAA is used to map a domain name to an IPv6 address. The IP6.ARPA domain is defined to look up a record given an IPv6 address.

- Finding Feature Information, page 55
- Prerequisites for Split DNS, page 56
- Restrictions for Split DNS, page 56
- Information About Split DNS, page 56
- How to Configure Split DNS, page 66
- Configuration Examples for Split DNS, page 84
- Additional References, page 88
- Feature Information for Split DNS, page 89
- Glossary, page 89

Finding Feature Information

Your software release may not support all the features documented in this module. For the latest caveats and feature information, see Bug Search Tool and the release notes for your platform and software release. To find information about the features documented in this module, and to see a list of the releases in which each feature is supported, see the feature information table at the end of this module.

Use Cisco Feature Navigator to find information about platform support and Cisco software image support. To access Cisco Feature Navigator, go to www.cisco.com/go/cfn. An account on Cisco.com is not required.

Prerequisites for Split DNS

No special equipment or software is needed to use the Split DNS feature. To use Split DNS to forward incoming DNS queries, you must have a client that issues DNS queries, a DNS caching name server on which the Split DNS features are to be configured, and a back-end DNS name server. Both of the DNS name server components reside in a Cisco device running the Cisco IOS DNS subsystem software. An example of this basic topology is illustrated in the figure below.

Restrictions for Split DNS

Data Link Layer Redirection

The DNS forwarding functionality provided by Split DNS to the DNS server subsystem of the Cisco IOS software is available only for DNS packets that are directed to one of the IP addresses of the device that serves as the DNS caching name server. Split DNS does not support processing of packets intercepted at the data link layer (Layer 2) and then redirected to the DNS caching name server.

Information About Split DNS

Split DNS Feature Overview

The Split DNS feature enables a Cisco device to answer DNS queries using the internal DNS hostname cache specified by the selected virtual DNS name server or, for queries that cannot be answered from the information in the hostname cache, direct queries to specific, back-end DNS servers. The virtual DNS name server is selected based on certain characteristics of each query. Split DNS commands are used to configure a customer premise equipment (CPE) device that serves as the DNS server and forwarder for queries from hosts and as the DNS server and resolver for queries originated by the device itself.

The following sections summarize Split DNS features:

Split DNS Use to Respond to DNS Queries Benefits

The following sections describe the primary Split DNS features:

Selection of Virtual DNS Caching Name Server Configurations

To configure a Split DNS environment, configure multiple DNS databases on the device and then configure the device to choose one of these virtual DNS server configurations whenever the device must respond to a DNS query by looking up or forwarding the query. The device that acts as the DNS forwarder or resolver is configured with multiple virtual DNS caching name server configurations, each associated with restrictions on the types of DNS queries that can be handled using that name server. The device can be configured to select a virtual forwarding or resolving DNS server configuration based on any combination of the following criteria:

- Query source port

- Query source interface Virtual Private Network (VPN) routing and forwarding (VRF) instance
- Query source authentication
- Query source IP address
- Query hostname

When the device must respond to a query, the Cisco IOS software selects a DNS name server by comparing the characteristics of the query to a list of name servers and their configured restrictions. After the appropriate name server is selected, the device addresses the query using the associated host table cache or forwarding parameters that are defined for that virtual name server.

Ability to Offload Internet Traffic from the Corporate DNS Server

When deployed in an enterprise network that supports many remote hosts with Internet VPN access to the central site, the Split DNS features of the Cisco IOS software enable the device to be configured to direct Internet queries to the Internet service provider (ISP) network, thus reducing the load on the corporate DNS server.

Compatibility with NAT and PAT

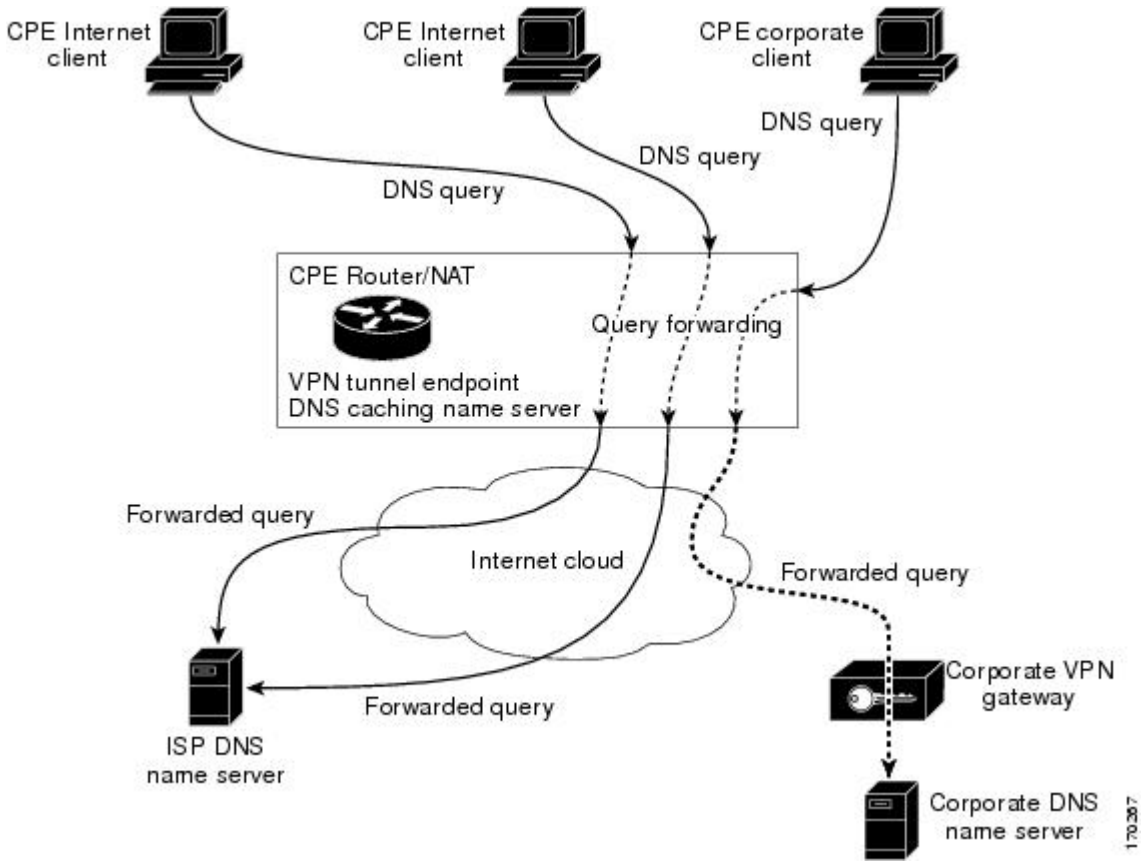
Split DNS is compatible with Network Address Translation (NAT) and Cisco IOS Port Address Translation (PAT) upstream interfaces. If NAT or PAT is enabled on the CPE device, DNS queries are translated (by address translation or port translation) to the appropriate destination address, such as an ISP DNS server or a corporate DNS server. When using split tunneling, the remote device routes the Internet-destined traffic directly, not forwarding it over the encrypted tunnel. With a remote client that uses split tunneling, it is possible for the device to direct DNS queries destined for the corporate DNS server to the pushed DNS server list from the central site if the tunnel is up and to direct DNS queries destined for the ISP DNS server to the outside public interface address if the tunnel is down.

Note Split tunneling requires additional security and firewall configuration to ensure the security of the remote site.

Split DNS Operation

A basic network topology for using Split DNS is illustrated in the figure below. The network diagram shows a CPE device that connects to both an ISP DNS name server and a corporate DNS name server. The diagram also shows three of the CPE client machines that access the device.

Figure 1: A Basic Network Topology for Split DNS



The following sections summarize the network activities in a basic Split DNS environment:

CPE Device Configuration

Configuration of the CPE device consists of defining DNS caching name server configurations and defining sets of rules for selecting one of the configurations to use for a given DNS query.

- Each DNS caching name server definition specifies an internal DNS hostname cache, DNS forwarding parameters, and DNS resolving parameters.
- Each set of configuration-selection rules consist of a list of name server configurations, with usage restrictions attached to each configuration in the list. The device can be configured with a default set of selection rules, and any device interface can be configured to use a set of selection rules.

DNS Query Issued by a CPE Client

The CPE client can issue DNS queries that request access to the Internet or to the corporate site. The basic network topology in the figure above shows a CPE device that receives incoming DNS queries from three clients, through interfaces that are enabled with NAT. The three client machines represent typical users of a corporate network:

- PC of a remote teleworker accessing noncorporate Internet sites
- Home PC that is being used by a family member of a home teleworker
- PC of a worker at the corporate site

The clients access the corporate network through a VPN tunnel that originates at the corporate VPN gateway and terminates in the CPE device.

Note The advantage of establishing the VPN tunnel from the corporate access system to the CPE device (rather than the endpoint client system) is that every other computer on the home LAN can also use the same tunnel, making it unnecessary to establish multiple tunnels (one for each system). In addition, the client system end user can use the tunnel when accessing corporate systems, without having to explicitly bring the tunnel up and down each time.

Virtual DNS Name Server Selection

Given an incoming DNS query, the Cisco IOS software uses either the default selection rules or the interface-specific selection rules (depending on the interface on which the query arrived) to select one of the DNS name server configurations in the list. To make the selection, the Cisco IOS software matches the query characteristics to the usage restrictions for each DNS name server configuration in the list. The selected configuration specifies both a host table cache and forwarding parameters, and the device uses this information to handle the query.

Response to the Client-issued DNS Query

The device handles the DNS query using the parameters specified by the selected DNS name server configuration:

1. If the query can be answered using the information in the internal DNS hostname cache specified by the selected virtual DNS name server, the device responds to the query.
2. If the query cannot be answered from the information in the hostname cache but DNS forwarding is enabled for the selected virtual DNS name server, the device sends the query to each of the configured DNS forwarders.
3. If no DNS forwarders are configured for the selected configuration, the device forwards the query using the name servers configured for the virtual DNS name server. For the three client machines (shown in the figure above) that request Internet access or access to the corporate site, the CPE device can forward those DNS queries to the appropriate DNS servers as follows:
   - An Internet access request from the PC of the remote teleworker would be forwarded to the ISP DNS name server.

- Similarly, an Internet access request from the PC of the family member of the home teleworker also would be forwarded to the ISP DNS name server.
- A DNS request for access to the corporate site from a worker, though, would be forwarded to the corporate DNS name server.

4 If no domain name servers are configured for the virtual DNS name server, the device forwards the query to the limited broadcast address (255.255.255.255) so that the query is received by all hosts on the local network segment but not forwarded by devices.

DNS Views

A DNS view is a set of parameters that specify how to handle a DNS query. A DNS view defines the following information:

- Association with a VRF
- Option to write to system message logging (syslog) output each time the view is used
- Parameters for resolving internally generated DNS queries
- Parameters for forwarding incoming DNS queries
- Internal host table for answering queries or caching DNS responses

Note The maximum number of DNS views and view lists supported is not specifically limited but is dependent on the amount of memory on the Cisco device. Configuring a larger number of DNS views and view lists uses more device memory, and configuring a larger number of views in the view lists uses more device processor time. For optimum performance, configure no more views and view list members than needed to support your Split DNS query forwarding or query resolution needs.

The following sections describe DNS views in further detail.

View Use Is Restricted to Queries from the Associated VRF

A DNS view is always associated with a VRF, whether it is the global VRF (the VRF whose name is a NULL string) or a named VRF. The purpose of this association is to limit the use of the view to handling DNS queries that arrive on an incoming interface matches a particular VRF:

- The global VRF is the default VRF that contains routing information for the global IP address space of the provider network. Therefore, a DNS view that is associated with the global VRF can be used only to handle DNS queries that arrive on an incoming interface in the global address space.
- A named VRF contains routing information for a VPN instance on a device in the provider network. A DNS view that is associated with a named VRF can be used only to handle DNS queries that arrive on an incoming interface that matches the VRF with which the view is associated.

Note Additional restrictions (described in "DNS View Lists") can be placed on a view after it has been defined. Also, a single view can be referenced multiple times, with different restrictions added in each case. However, because the association of a DNS view with a VRF is specified in the DNS view definition, the VRF-specific view-use limitation is a characteristic of the DNS view definition itself and cannot be separated from the view.

Parameters for Resolving Internally Generated DNS Queries

The following parameters define how to resolve internally generated DNS queries:

- Domain lookup--Enabling or disabling of DNS lookup to resolve hostnames for internally generated queries.
- Default domain name--Default domain to append to hostnames without a dot.
- Domain search list--List of domain names to try for hostnames without a dot.
- Domain name for multicast lookups--IP address to use for multicast address lookups.
- Lookup timeout--Time (in seconds) to wait for a DNS response after sending or forwarding a query.
- Lookup retries--Number of retries when sending or forwarding a query.
- Domain name servers--List of name servers to use to resolve domain names for internally generated queries.
- Resolver source interface--Source interface to use to resolve domain names for internally generated queries.
- Round-robin rotation of IP addresses--Enabling or disabling of the use of a different IP address associated with the domain name in cache each time hostnames are looked up.

Parameters for Forwarding Incoming DNS Queries

The following parameters define how to forward incoming DNS queries:

- Forwarding of queries--Enabling or disabling of forwarding of incoming DNS queries.
- Forwarder addresses--List of IP addresses to use to forward incoming DNS queries.
- Forwarder source interface--Source interface to use to forward incoming DNS queries.

Sometimes, when a source interface is configured on a device with the split DNS feature to forward DNS queries, the device does not forward the DNS queries through the configured interface. Hence, consider the following points while forwarding the DNS queries using the source interface:

- DNS queries are forwarded to a broadcast address when a forwarding source interface is configured and the DNS forwarder is not configured.
- The source IP address of the forwarded query should be set to the primary IP address of the interface configured, using the **dns forwarding source-interface** *interface* command. If no such configuration exists, then the source IP address of the forwarded DNS query will be the primary IP address of the

outgoing interface. DNS forwarding should be done only when the source interface configured for the DNS forwarding is active.

- The source IP address of the DNS query for the DNS resolver functionality is set using the **domain resolver source-interface** *interface-type number* command. If there is no DNS address configured, then queries will be broadcasted to the defined source interface. DNS resolving should be done only when the source interface configured for the DNS resolving is active. See "Specifying a Source Interface to Forward DNS Queries" for the configuration steps.

DNS View Lists

A DNS view list is an ordered list of DNS views in which additional usage restrictions can be specified for any individual member in the list. The scope of these optional usage restrictions is limited to a specific member of a specific DNS view list. When the device must respond to a DNS query, the Cisco IOS software uses a DNS view list to select the DNS view that will be used to handle a DNS query.

Note The maximum number of DNS views and view lists supported is not specifically limited but is dependent on the amount of memory on the Cisco device. Configuring a larger number of DNS views and view lists uses more device memory, and configuring a larger number of views in the view lists uses more device processor time. For optimum performance, configure no more views and view list members than needed to support your Split DNS query forwarding or query resolution needs.

Order in Which to Check the Members of a DNS View List

When a DNS view list is used to select a DNS view for handling a given DNS query, the Cisco IOS software checks each member of the view list--in the order specified by the list--and selects the first view list member whose restrictions permit the view to be used with the query that needs to be handled.

Usage Restrictions Defined for a DNS View in the View List

A DNS view list member can be configured with usage restrictions defined using access control lists (ACLs) that specify rules for selecting that view list member based on the query hostname or the query source host IP address. The two types of ACLs supported by the Split DNS view list definition are described in "DNS Name Groups".

Note Multiple DNS view lists can be defined so that, for example, a given DNS view can be associated with different restrictions in each list. Also, different DNS view lists can include different DNS views.

Selection of the DNS View List

When the device that is acting as the DNS caching name server needs to respond to a DNS query, the Cisco IOS software uses a DNS view list to determine which DNS view can be used to handle the query:

- If the device is responding to an incoming query that arrives on an interface for which a DNS view list is configured, the interface-specific DNS view list is used.

- If the device is responding to an incoming query that arrives on an interface for which no specific DNS view list is configured, the default DNS view list is used.

If the device is responding to an internally generated query, no DNS view list is used to select a view; the global DNS view is used to handle the query.

The assignment of a DNS view list as the default or to an interface is described in "DNS View Groups".

Selection of a DNS View List Member

The view list members are compared, each in turn, to the characteristics of the DNS query that the device is responding to:

1. If the query is from a different VRF than the view, the view cannot be used to address the query, so the view-selection process moves on to the next member of the view list.
2. The specification of additional view-use restrictions is an optional setting for any view list member.

If the query list does not specify additional restrictions on the view, the view will be used to address the query, so the view-selection process is finished.

If the view list does specify additional restrictions on the view, the query is compared to those restrictions:

- If the query characteristics fail any view-use restriction, the view cannot be used to address the query, so the view-selection process moves on to the next member of the view list.
- If the query characteristics pass all the view-use restrictions, the view will be used to address the query. The view-selection process is finished.
- If the view-selection process reaches the end of the selected DNS view list without finding a view list member that can handle the query, the device discards the query.

The first DNS view list member that is found to have restrictions that match the query characteristics is used to handle the query.

DNS Name Groups

The Split DNS feature supports two types of ACLs that can be used to restrict the use of a DNS view. A DNS name list or a standard IP ACL (or both) can be applied to a DNS view list member to specify view-use restrictions in addition to the VRF-specific restriction that is a part of the view definition itself.

Note In this context, the term “group” is used to refer to the specification of a DNS name list or a standard IP ACL as a usage restriction on a view list member.

DNS View Usage Restrictions Based on the Query Hostname

A DNS name list is a named set of hostname pattern-matching rules, with each rule specifying the type of action to be performed if a query hostname matches the text string pattern in the rule. In order for a query hostname to match a name list, the hostname must match a rule that explicitly permits a matching pattern but the hostname cannot match any rules that explicitly deny a matching pattern.

DNS View Usage Restrictions Based on the Query Source IP Address

A standard IP ACL is a numbered or named set of host IP address-matching rules, with each rule specifying the type of action to be performed if an IP address matches the text string pattern in the rule. The Split DNS feature supports the use of a standard ACL as a view-use restriction based on the query source IP address. In order for a source IP address to match a name list, the IP address must match a rule that explicitly permits a matching pattern but the IP address cannot match any rules that explicitly deny a matching pattern.

DNS View Groups

The Split DNS feature provides two ways to specify the DNS view list that the Cisco IOS software is to use to select the DNS view that will be used to handle an incoming DNS query. For a query that arrives on an interface that is configured to use a particular DNS view list, the interface-specific DNS view list is used. Otherwise, the default DNS view list is used.

Note In this context, the term "group" refers to the specification of a DNS view list as an interface-specific DNS view list or the default view list for the device.

Interface-specific View Lists

A DNS view list can be attached to a device interface. When an incoming DNS query arrives on that interface, the Cisco IOS software uses that view list to select a DNS view to use to handle the query.

Default DNS View List

A DNS view list can be configured as the default DNS view list for the device. When an incoming DNS query arrives on an interface that is not configured to use a specific view list, the Cisco IOS software uses the default view list to select the DNS view to use to handle the query.

Device Response to DNS Queries in a Split DNS Environment

By introducing support of DNS views--and the ability to configure the device to select from a list of appropriate views for a given DNS query--the Split DNS feature enables different hosts and subsystems to use different virtual DNS caching name servers, each with their own, separate DNS cache and each accessible from a single device that acts as the DNS forwarder and resolver. Thus, each DNS view defines a different DNS database on a single device. Furthermore, because the Split DNS feature separates the configuration of DNS query forwarding and resolving parameters, it is a simple matter to configure the device to respond more freely to queries from internal clients while limiting response to queries from external clients.

If the device receives a query other than a broadcast, it forwards the query as a broadcast under the VRF as defined in the interface view:

- If a device is acting as a forwarder.
- If at least one global name-server is configured.
- If the view to be used to service this query does not contain any of the following commands:
  - **dns forwarder** [**vrf** *vrf-name*] *forwarder-ip-address*
  - **dns forwarding source-interface** *interface*

- **domain name-server** [**vrf** *vrf-name*] *name-server-ip-address*
- **domain resolver source-interface** *interface-type number*

See "Specifying a DNS View List for a Device Interface" to specify a DNS view list for a particular device interface.

The following sections provide detailed descriptions of how the device responds to DNS queries in a Split DNS environment.

Response to Incoming DNS Queries per the Forwarding Parameters of the Selected DNS View

Given an incoming DNS query, the Cisco IOS software uses the DNS view list configured for that interface to select the DNS view list to use to handle the query. If no view list is configured for the interface, the default DNS view list is used instead.

Using the configured or default view list, the device software selects the first view list member that is associated with the same VRF as the query and whose usage restrictions match the query characteristics. After the DNS view is selected, the device handles the query according to the parameters configured in the selected view.

1. The device uses the DNS view list that is specified for the interface on which the DNS query arrives:
    1. If a DNS view list is attached to the interface, the device uses the specified DNS view list.
    2. If no DNS view list is attached to the interface, the device uses the default DNS view list.
2. The device uses the DNS view list to select a DNS view to use to address the query. Each view list member is checked, in the order defined by the view list, as follows:
    1. If the view list member is associated with a different VRF from that of the incoming interface for the DNS query that needs to be resolved, the view-selection process moves on to the next member of the view list.
    2. If all the usage restrictions on the view list member match the other characteristics of the DNS query to be resolved, the view is selected to handle the query.

Otherwise, the view-selection process moves on to the next member of the view list.

If no member of the default DNS view list is qualified to address the query, the device does nothing further with the query.

1. The device attempts to respond to the query using the parameters specified by the selected DNS view:
    1. The Cisco IOS software looks in the hostname cache associated with the view. If the query can be answered from that information, the device responds to the query.
    2. If the query cannot be answered using the hostname cache, the Cisco IOS software checks whether the DNS forwarding of queries is enabled for the view. If DNS forwarding is enabled, the device sends the query to each of the configured DNS forwarders.
    3. If no DNS forwarders are configured for the view, the device forwards the query using the configured domain name servers.
    4. If no domain name servers are configured for the view, the device forwards incoming DNS queries to the limited broadcast address (255.255.255.255) so that the queries are received by all hosts on the local network segment but not forwarded by devices.

Response to Internally Generated DNS Queries per the Resolving Parameters of the Default Global DNS View

Given an internally generated DNS query to resolve, the Cisco IOS software uses the default DNS view to handle the query:

- When a hostname must be resolved for a query that does not specify a VRF, the device uses the unnamed DNS view associated with the global VRF (the default VRF that contains routing information for the global IP address space of the provider network).
- When a hostname must be resolved for a Cisco IOS command that specifies a VRF to use, the device uses the unnamed DNS view associated with that VRF.

The device attempts to respond to the query using the DNS resolving parameters specified by that view:

1. If the query specifies an unqualified hostname, the Cisco IOS software completes the hostname using the domain name list or the default domain specified by the view.
2. The Cisco IOS software looks in the hostname cache associated with the view. If the query can be answered from that information, the device responds to the query.
3. Otherwise, because the query cannot be answered using the hostname cache, the Cisco IOS software checks whether the DNS forwarding of queries is enabled for the view. If so, the device sends the query to each of the configured name servers, using the timeout period and number of retries specified for the view.
4. Otherwise, the device does not respond to the query.

How to Configure Split DNS

Enabling Split DNS Debugging Output

Enabling a Split DNS **debug** command enables output to be written at every occurrence of a DNS name list event, a DNS view event, or a DNS view list event. The device continues to generate such output until you enter the corresponding **no debug** command. You can use the output from the Split DNS **debug** commands to diagnose and resolve internetworking problems associated with Split DNS operations.

Note By default, the network server sends the output from the **debug** commands to the console. Sending output to a terminal (virtual console) produces less overhead than sending it to the console. Use the **terminal monitor** privileged EXEC command to send output to a terminal. For more information about redirecting **debug** command output, see the "Using Debug Commands" chapter of the *Cisco IOS Debug Command Reference* .

A DNS name list event can be of any of the following:

- The addition or removal of a DNS name list entry (a hostname pattern and action to perform on an incoming DNS query for a hostname that matches the pattern).
- The removal of a DNS name list.

A DNS view event can be any of the following:

- The addition or removal of a DNS view definition.
- The addition or removal of a DNS forwarding name server setting for a DNS view.
- The addition or removal of a DNS resolver setting for a DNS view.
- The enabling or disabling of logging of a syslog message each time a DNS view is used.

A DNS view list event can be any of the following:

- The addition or removal of a DNS view list definition.
- The addition or removal of a DNS view list member (a DNS view and the relative order in which it is to be checked in the view list) to or from a DNS view list.
- The setting or clearing of a DNS view list assignment as the default view list for the device or to a specific interface on the device.

Perform this optional task if you want to enable the writing of an event message to syslog output for DNS name list events, view events, or view list events:

SUMMARY STEPS

1. **enable**
2. **debug ip dns name-list**
3. **debug ip dns view**
4. **debug ip dns view-list**
5. **show debugging**

DETAILED STEPS

	Command or Action	Purpose
Step 1	**enable** **Example:** `Device> enable`	Enables privileged EXEC mode. • Enter your password if prompted.
Step 2	**debug ip dns name-list** **Example:** `Device# debug ip dns name-list`	(Optional) Enables the writing of DNS name list event messages. • Debugging output for DNS name lists is disabled by default. • To disable debugging output for DNS name list events, use the **no** form of this command.
Step 3	**debug ip dns view** **Example:** `Device# debug ip dns view`	(Optional) Enables the writing of DNS view event messages. • Debugging output for DNS views is disabled by default. • To disable debugging output for DNS view events, use the **no** form of this command.

	Command or Action	Purpose
Step 4	**debug ip dns view-list** **Example:** `Device# debug ip dns view-list`	(Optional) Enables the writing of DNS view list event messages. • Debugging output for DNS view lists is disabled by default. • To disable debugging output for DNS view list events, use the **no** form of this command.
Step 5	**show debugging** **Example:** `Device# show debugging`	Displays the state of each debugging option.

Defining a DNS Name List

Perform this optional task if you need to define a DNS name list. A DNS name list is a list of hostname pattern-matching rules that could be used as an optional usage restriction on a DNS view list member.

SUMMARY STEPS

1. **enable**
2. **configure terminal**
3. **no ip dns name-list** *name-list-number* [{**deny** | **permit**} *pattern*]
4. **ip dns name-list** *name-list-number* {**deny** | **permit**} *pattern*
5. **exit**
6. **show ip dns name-list** [*name-list-number*]

DETAILED STEPS

	Command or Action	Purpose
Step 1	**enable** **Example:** `Device> enable`	Enables privileged EXEC mode. • Enter your password if prompted.
Step 2	**configure terminal** **Example:** `Device# configure terminal`	Enters global configuration mode.

	Command or Action	Purpose
Step 3	**no ip dns name-list** *name-list-number* [{**deny** | **permit**} *pattern*] **Example:** `Device(config)# no ip dns name-list 500`	(Optional) Clears any previously defined DNS name list. • To clear only an entry in the list, specify the **deny** or **permit** clause. • To clear the entire list, omit any clauses.
Step 4	**ip dns name-list** *name-list-number* {**deny** | **permit**} *pattern* **Example:** `Device(config)# ip dns name-list 500 deny .*.example.com`	Creates a new entry in the specified DNS name list. • The *pattern* argument specifies a regular expression that will be compared to the query hostname. For a detailed description of regular expressions and regular expression pattern-matching characters, see the appendix titled "Regular Expressions" in the *Cisco IOS Terminal Services Configuration Guide* . • The **deny** keyword specifies that any name matching the specified pattern immediately terminates matching the name list with a negative result. The **permit** keyword specifies that any name matching the specified pattern immediately terminates matching the name list with a positive result. • Enter this command multiple times as needed to create multiple deny and permit clauses. • To apply a DNS name list to a DNS view list member, use the **restrict name-group** command.
Step 5	**exit** **Example:** `Device(config)# exit`	Exits global configuration mode.
Step 6	**show ip dns name-list** [*name-list-number*] **Example:** `Device# show ip dns name-list`	Displays a particular DNS name list or all configured name lists.

Defining a DNS View

Perform this task to define a DNS view. A DNS view definition can be used to respond to either an incoming DNS query or an internally generated DNS query.

SUMMARY STEPS

1. **enable**
2. **configure terminal**
3. **ip dns view** [**vrf** *vrf-name*] {**default** | *view-name*}
4. [**no**] **logging**
5. [**no**] **domain lookup**
6. Do one of the following:
   - **domain name** *domain-name*
   - **domain list** *domain- name*
7. Do one of the following:
   - **domain name-server** [**vrf** *vrf-name*] *name-server-ip-address*
   - **domain name-server interface** *interface*
8. **domain multicast** *domain-name*
9. **domain retry** *number*
10. **domain timeout** *seconds*
11. [**no**] **dns forwarding**
12. **dns forwarder** [**vrf** *vrf-name*] *forwarder-ip-address*
13. **dns forwarding source-interface** *interface*
14. **end**
15. **show ip dns view** [**vrf** *vrf-name*] [**default** | *view-name*]

DETAILED STEPS

	Command or Action	Purpose
Step 1	**enable** **Example:** `Device> enable`	Enables privileged EXEC mode. • Enter your password if prompted.
Step 2	**configure terminal** **Example:** `Device# configure terminal`	Enters global configuration mode.
Step 3	**ip dns view** [**vrf** *vrf-name*] {**default** | *view-name*}	Defines a DNS view and enters DNS view configuration mode.

	Command or Action	Purpose
	Example: `Device(config)# ip dns view vrf vpn101 user3`	
Step 4	[**no**] **logging** **Example:** `Device(cfg-dns-view)# logging`	(Optional) Enables or disables logging of a syslog message each time the DNS view is used. **Note** View-specific event logging is disabled by default.
Step 5	[**no**] **domain lookup** **Example:** `Device(cfg-dns-view)# domain lookup`	(Optional) Enables or disables DNS-based hostname-to-address translation for internally generated DNS queries handled using the DNS view. **Note** The domain lookup capability is enabled by default.

Step 6	Do one of the following:
	• **domain name** *domain-name*
	• **domain list** *domain- name*
	Example:
	`Device(cfg-dns-view)# domain name example.com`
	Example:
	`Device(cfg-dns-view)# domain list example1.com`
		(Optional) Defines a default domain name to be used by this DNS view to complete unqualified hostnames when addressing DNS queries.
		or
		(Optional) Defines a list of domain names to be used by this DNS view to complete unqualified hostnames when addressing DNS queries.
		• The device attempts to respond to the query using the parameters specified by the selected DNS view. First, the Cisco IOS software looks in the hostname cache associated with the view. If the query can be answered from that information, the device responds to the query. Otherwise, because the query cannot be answered using the hostname cache, the device forwards the query using the configured domain name servers.
		• If the device is using this view to handle a DNS query for an unqualified hostname and domain lookup is enabled for the view, the Cisco IOS software appends a domain name (either a domain name from the domain name list or the default domain name) in order to perform any of the following activities:
		  • Looking up the hostname in the name server cache.
		  • Forwarded the query to other name servers (whether to the hosts specified as DNS forwarders in the selected view or to the limited broadcast address).
		• You can specify a single, default domain name, an ordered list of domain names, or both. However, the default domain name is used only if the domain list is empty.

	Command or Action	Purpose

Step 7	Do one of the following:
	• **domain name-server** [**vrf** *vrf-name*] *name-server-ip-address*
	• **domain name-server interface** *interface*
	Example:
	`Device(cfg-dns-view)# domain name-server 192.168.2.124`
	Example:
	`Device(cfg-dns-view)# domain name-server interface FastEthernet0/1`
		(Optional) Defines a list of name servers to be used by this DNS view to resolve internally generated DNS queries. The IP address of the name server can be an IPv4 or IPv6 address, and the IP address can be associated with a Virtual Private Network (VPN) routing and forwarding (VRF) instance.
		or
		(Optional) Defines an interface on which to acquire (through DHCP or PPP interaction on the interface) the IP address of a DNS server to add to the list of DNS name servers to be used by this DNS view to resolve internally generated DNS queries.
		• If both of these commands are configured, DHCP or PPP interaction on the interface causes another IP address to be added to the list.

Step 8	**domain multicast** *domain-name* **Example:** `Device(cfg-dns-view)# domain multicast www.example8.com`	(Optional) Specifies the IP address to use for multicast lookups handled using the DNS view.
Step 9	**domain retry** *number* **Example:** `Device(cfg-dns-view)# domain retry 4`	(Optional) Defines the number of times to perform a retry when using this DNS view to send or forward DNS queries. **Note** The number of retries is 2 by default.
Step 10	**domain timeout** *seconds* **Example:** `Device(cfg-dns-view)# domain timeout 5`	(Optional) Defines the number of seconds to wait for a response to a DNS query sent or forwarded when using this DNS view. **Note** The time to wait is 3 seconds by default.
Step 11	[**no**] **dns forwarding** **Example:** `Device(cfg-dns-view)# dns forwarding`	(Optional) Enables or disables forwarding of incoming DNS queries handled using the DNS view. **Note** The query forwarding capability is enabled by default.
Step 12	**dns forwarder** [**vrf** *vrf-name*] *forwarder-ip-address*	Defines a list of name servers to be used by this DNS view to forward incoming DNS queries.

	Command or Action	Purpose
	Example: `Device(cfg-dns-view)# dns forwarder 192.168.3.240`	• The forwarder IP address can be an IPv4 or IPv6 address. • If no forwarding name servers are defined, then the configured list of domain name servers is used instead. • If no name servers are configured either, then queries are forwarded to the limited broadcast address.
Step 13	**dns forwarding source-interface** *interface* **Example:** `Device(cfg-dns-view)# dns forwarding source-interface FastEthernet0/0`	Defines the interface on which to forward queries when this DNS view is used.
Step 14	**end** **Example:** `Device(cfg-dns-view)# end`	Returns to privileged EXEC mode.
Step 15	**show ip dns view** [**vrf** *vrf-name*] [**default** | *view-name*] **Example:** `Device# show ip dns view vrf vpn101 user3`	Displays information about a particular DNS view, a group of views (with the same view name or associated with the same VRF), or all configured DNS views.

Defining Static Entries in the Hostname Cache for a DNS View

It is easier to refer to network devices by symbolic names rather than numerical addresses (services such as Telnet can use hostnames or addresses). Hostnames and IP addresses can be associated with one another through static or dynamic means. Manually assigning hostnames-to-address mappings is useful when dynamic mapping is not available.

Perform this optional task if you need to define static entries in the DNS hostname cache for a DNS view.

SUMMARY STEPS

1. **enable**
2. **clear host** [**view** *view-name* | **vrf** *vrf-name* | **all**] {*hostname* | *****}
3. **configure terminal**
4. **ip host** [**vrf** *vrf-name*] [**view** *view-name*] *hostname* {*ip-address1* [*ip-address2...ip-address8*] | **additional** *ip-address9* [*ip-address10...ip-addressn*]}
5. **exit**
6. **show hosts** [**vrf** *vrf-name*] [**view** *view-name*] [**all** | *hostname*] [**summary**]

DETAILED STEPS

	Command or Action	Purpose
Step 1	**enable** **Example:** `Device> enable`	Enables privileged EXEC mode. • Enter your password if prompted.
Step 2	**clear host** [**view** *view-name* | **vrf** *vrf-name* | **all**] {*hostname* | *****} **Example:** `Device# clear host all *`	(Optional) Removes static hostname-to-address mappings from the hostname cache for the specified DNS view or all configured views. • Use the **view** keyword and *view-name* argument to specify the DNS view whose hostname cache is to be cleared. Default is the default DNS view associated with the specified or global VRF. • Use the **vrf** keyword and *vrf-name* argument to specify the VRF associated with the DNS view whose hostname cache is to be cleared. Default is the global VRF (that is, the VRF whose name is a NULL string) with the specified or default DNS view. • Use the **all** keyword to specify that hostname-to-address mappings are to be deleted from the hostname cache of every configured DNS view. • Use the *hostname* argument to specify the name of the host for which hostname-to-address mappings are to be deleted from the specified hostname cache. • Use the ***** keyword to specify that all the hostname-to-address mappings are to be deleted from the specified hostname cache.
Step 3	**configure terminal** **Example:** `Device# configure terminal`	Enters global configuration mode.
Step 4	**ip host** [**vrf** *vrf-name*] [**view** *view-name*] *hostname* {*ip-address1* [*ip-address2...ip-address8*] | **additional** *ip-address9* [*ip-address10...ip-addressn*]} **Example:** `Device(config)# ip host vrf vpn101 view user3 www.example.com 192.168.2.111 2001:DB8:1::1`	Defines static hostname-to-address mappings in the DNS hostname cache for a DNS view. • More than one DNS view can be associated with a VRF. To uniquely identify a DNS view, specify both the view name and the VRF with which it is associated. • The host IP address can be an IPv4 or IPv6 address. • Use the *hostname* argument to specify the name of the host for which hostname-to-address mappings are to be added to the specified hostname cache. • To bind more than eight addresses to a hostname, you can use the ip host command again and use the **additional** keyword.

	Command or Action	Purpose
Step 5	**exit** **Example:** `Device(config)# exit`	Exits global configuration mode.

Step 6	**show hosts** [**vrf** *vrf-name*] [**view** *view-name*] [**all** | *hostname*] [**summary**]
	
	Example:
	
	`Device# show hosts vrf vpn101 view user3`
	`www.example.com`
		(Optional) Displays the default domain name, the style of name lookup service, a list of name server hosts, and the cached list of hostnames and addresses specific to a particular DNS view or for all configured DNS views.
		• More than one DNS view can be associated with a VRF. To uniquely identify a DNS view, specify both the view name and the VRF with which it is associated.
		• Use the **all** keyword if the specified hostname cache information is to be displayed for all configured DNS views.
		• Use the *hostname* argument if the specified name cache information displayed is to be limited to entries for a particular hostname.

Defining a DNS View List

Perform this task to define an ordered list of DNS views with optional, additional usage restrictions for each view list member. The device uses a DNS view list to select the DNS view that will be used to handle a DNS query.

SUMMARY STEPS

1. **enable**
2. **configure terminal**
3. **ip dns view-list** *view-list-name*
4. **view** [**vrf** *vrf-name*] {**default** | *view-name*} *order-number*
5. **restrict name-group** *name-list-number*
6. **restrict source access-group** *acl-number*
7. **exit**
8. **end**
9. **show ip dns view-list** *view-list-name*

DETAILED STEPS

	Command or Action	Purpose
Step 1	**enable**	Enables privileged EXEC mode.

	Command or Action	Purpose
	Example: `Device> enable`	• Enter your password if prompted.
Step 2	**configure terminal** **Example:** `Device# configure terminal`	Enters global configuration mode.
Step 3	**ip dns view-list** *view-list-name* **Example:** `Device(config)# ip dns view-list userlist5`	Defines a DNS view list and enters DNS view list configuration mode.
Step 4	**view** [**vrf** *vrf-name*] {**default** | *view-name*} *order-number* **Example:** `Device(cfg-dns-view-list)# view vrf vpn101 user5 10`	Defines a DNS view list member and enters DNS view list member configuration mode.
Step 5	**restrict name-group** *name-list-number* **Example:** `Device(cfg-dns-view-list-member)# restrict name-group 500`	(Optional) Specifies that this DNS view list member cannot be used to respond to a DNS query unless the query hostname matches a permit clause in the specified DNS name list and none of the deny clauses. • To define a DNS name list entry, use the **ip dns name-list** command.
Step 6	**restrict source access-group** *acl-number* **Example:** `Device(cfg-dns-view-list-member)# restrict access-group 99`	(Optional) Specifies that this DNS view list member cannot be used to respond to a DNS query unless the source IP address of the DNS query matches the specified standard ACL. • To define a standard ACL entry, use the **access-list** command.
Step 7	**exit** **Example:** `Device(cfg-dns-view-list-member)# exit`	Exits DNS view list member configuration mode. • To add another view list member to the list, go to Step 4.
Step 8	**end** **Example:** `Device(cfg-dns-view-list)# end`	Returns to privileged EXEC mode.

	Command or Action	Purpose
Step 9	**show ip dns view-list** *view-list-name* **Example:** `Device# show ip dns view-list userlist5`	Displays information about a particular DNS view list or all configured DNS view lists.

Modifying a DNS View List

To provide for efficient management of the order of the members in a view list, each view list member definition includes the specification of the position of that member within the list. That is, the order of the members within a view list is defined by explicit specification of position values rather than by the order in which the individual members are added to the list. This enables you to perform either of the following tasks without having to remove all the view list members and then redefine the view list membership in the desired order:

Adding a Member to a DNS View List Already in Use

Perform this optional task if you need to add another member to a DNS view list that is already in use.

For example, suppose the DNS view list named userlist5 is already defined and in use as a default view list or as an interface-specific view list. Assume that the list consists of the following members:

- DNS view user1 with position number 10
- DNS view user2 with position number 20
- DNS view user3 with position number 30

If you need to add DNS view user4 as the second member of the list, add that view to the list with a position number value from 11 to 19. You do not need to remove the three existing members and then add all four members to the list in the desired order.

SUMMARY STEPS

1. **enable**
2. **show ip dns view-list** *view-list-name*
3. **configure terminal**
4. **ip dns view-list** *view-list-name*
5. **view** [**vrf** *vrf-name*] {**default** | *view-name*} *order-number*
6. **end**
7. **show ip dns view-list** *view-list-name*

DETAILED STEPS

	Command or Action	Purpose
Step 1	**enable** **Example:** `Device> enable`	Enables privileged EXEC mode. • Enter your password if prompted.
Step 2	**show ip dns view-list** *view-list-name* **Example:** `Device# show ip dns view-list userlist5`	Displays information about a particular DNS view list or all configured DNS view lists.
Step 3	**configure terminal** **Example:** `Device# configure terminal`	Enters global configuration mode.
Step 4	**ip dns view-list** *view-list-name* **Example:** `Device(config)# ip dns view-list userlist5`	Defines a DNS view list and enters DNS view list configuration mode.
Step 5	**view** [**vrf** *vrf-name*] {**default** | *view-name*} *order-number* **Example:** `Device(cfg-dns-view-list)# view user4 15`	Defines a DNS view list member and enters DNS view list member configuration mode.
Step 6	**end** **Example:** `Device(cfg-dns-view-list-member)# end`	Returns to privileged EXEC mode.
Step 7	**show ip dns view-list** *view-list-name* **Example:** `Device# show ip dns view-list userlist5`	Displays information about a particular DNS view list or all configured DNS view lists.

Changing the Order of the Members of a DNS View List Already in Use

Perform this optional task if you need to change the order of the members of a DNS view list that is already in use.

For example, suppose the DNS view list named userlist5 is already defined and in use as a default view list or as an interface-specific view list. Assume that the list consists of the following members:

- DNS view user1 with position number 10
- DNS view user2 with position number 20
- DNS view user3 with position number 30

If you want to move DNS view user1 to the end of the list, remove that view from the list and then add it back to the list with a position number value greater than 30. You do not need to remove the three existing members and then add the members back to the list in the desired order.

SUMMARY STEPS

1. **enable**
2. **show ip dns view-list** *view-list-name*
3. **configure terminal**
4. **ip dns view-list** *view-list-name*
5. **no view** [**vrf** *vrf-name*] {**default** | *view-name*} *order-number*
6. **view** [**vrf** *vrf-name*] {**default** | *view-name*} *order-number*
7. **end**
8. **show ip dns view-list** *view-list-name*

DETAILED STEPS

	Command or Action	Purpose
Step 1	**enable** **Example:** `Device> enable`	Enables privileged EXEC mode. • Enter your password if prompted.
Step 2	**show ip dns view-list** *view-list-name* **Example:** `Device# show ip dns view-list userlist5`	Displays information about a particular DNS view list or all configured DNS view lists.
Step 3	**configure terminal** **Example:** `Device# configure terminal`	Enters global configuration mode.
Step 4	**ip dns view-list** *view-list-name* **Example:** `Device(config)# ip dns view-list userlist5`	Defines a DNS view list and enters DNS view list configuration mode.

	Command or Action	Purpose
Step 5	**no view** [**vrf** *vrf-name*] {**default** | *view-name*} *order-number* **Example:** `Device(cfg-dns-view-list)# no view user1 10`	Removes a DNS view list member from the list.
Step 6	**view** [**vrf** *vrf-name*] {**default** | *view-name*} *order-number* **Example:** `Device(cfg-dns-view-list)# view user1 40`	Defines a DNS view list member and enters DNS view list member configuration mode.
Step 7	**end** **Example:** `Device(cfg-dns-view-list-member)# end`	Returns to privileged EXEC mode.
Step 8	**show ip dns view-list** *view-list-name* **Example:** `Device# show ip dns view-list userlist5`	Displays information about a particular DNS view list or all configured DNS view lists.

Specifying the Default DNS View List for the DNS Server of the Device

Perform this task to specify the default DNS view list for the device's DNS server. The device uses the default DNS view list to select a DNS view to use to handle an incoming DNS query that arrives on an interface for which no interface-specific DNS view list has been defined.

SUMMARY STEPS

1. **enable**
2. **configure terminal**
3. **ip dns server view-group** *name-list-number*
4. **exit**
5. **show running-config**

DETAILED STEPS

	Command or Action	Purpose
Step 1	**enable**	Enables privileged EXEC mode.

	Command or Action	Purpose
	Example: `Device> enable`	• Enter your password if prompted.
Step 2	**configure terminal** **Example:** `Device# configure terminal`	Enters global configuration mode.
Step 3	**ip dns server view-group** *name-list-number* **Example:** `Device(config)# ip dns server view-group 500`	Configures the default DNS view list for the device's DNS server.
Step 4	**exit** **Example:** `Device(config)# exit`	Exits global configuration mode.
Step 5	**show running-config** **Example:** `Device# show running-config`	Displays information about how DNS view lists are applied. The default DNS view list, if configured, is listed in the default DNS view information as the argument for the **ip dns server view-group** command.

Specifying a DNS View List for a Device Interface

Perform this optional task if you need to specify a DNS view list for a particular device interface. The device uses that view list to select a DNS view to use to handle a DNS query that arrives on that interface.

SUMMARY STEPS

1. **enable**
2. **configure terminal**
3. **interface** *interface*
4. **ip dns view-group** *view-list-name*
5. **end**
6. **show running-config**

DETAILED STEPS

	Command or Action	Purpose
Step 1	**enable** **Example:** `Device> enable`	Enables privileged EXEC mode. • Enter your password if prompted.
Step 2	**configure terminal** **Example:** `Device# configure terminal`	Enters global configuration mode.
Step 3	**interface** *interface* **Example:** `Device(config)# interface ATM2/0`	Configures an interface type and enter interface configuration mode so that the specific interface can be configured.
Step 4	**ip dns view-group** *view-list-name* **Example:** `Device(config-if)# ip dns view-group userlist5`	Configures the DNS view list for this interface on the device.
Step 5	**end** **Example:** `Device(config-if)# end`	Returns to privileged EXEC mode.
Step 6	**show running-config** **Example:** `Device# show running-config`	Displays information about how DNS view lists are applied. Any DNS view lists attached to interfaces are listed in the information for each individual interface, as the argument for the **ip dns view-group** command.

Specifying a Source Interface to Forward DNS Queries

Perform this optional task if you need to specify a source interface to forward the DNS queries.

SUMMARY STEPS

1. **enable**
2. **configure terminal**
3. **ip dns view** [**vrf** *vrf-name*] {**default** | *view-name*}
4. **domain resolver source-interface** *interface-type number*
5. **end**

DETAILED STEPS

	Command or Action	Purpose
Step 1	**enable** **Example:** `Device> enable`	Enables privileged EXEC mode. • Enter your password if prompted.
Step 2	**configure terminal** **Example:** `Device# configure terminal`	Enters global configuration mode.
Step 3	**ip dns view** [**vrf** *vrf-name*] {**default**	*view-name*} **Example:** `Device(config)# ip dns view vrf vpn32 user3`	Creates the DNS view of the specified name associated with the specified VRF instance and then enters DNS view configuration mode.
Step 4	**domain resolver source-interface** *interface-type number* **Example:** `Device(cfg-dns-view)# domain resolver source-interface fastethernet 0/0`	Sets the source IP address of the DNS queries for the DNS resolver functionality.
Step 5	**end** **Example:** `Device(config-if)# end`	(Optional) Returns to privileged EXEC mode.

Configuration Examples for Split DNS

Example: Split DNS View Limited to Queries from a Specific VRF

The following example shows how to define two different VRFs and then define two different DNS views that are associated with those VRFs:

```
ip vrf vpn101
 description VRF vpn101 for example purposes
 rd 10:112
 exit
!
ip vrf vpn102
 description VRF vpn102 for example purposes
 rd 10:128
 exit
!
ip dns view vrf vpn101
 .
 .
 .
exit
!
ip dns view vrf vpn102 user1
 .
 .
 .
exit
```

The two DNS views are both named user1, but each view is associated with a different VRF.

- The default DNS view associated with VRF vpn101 is limited to handling DNS queries from VRF vpn101 only. This view will be used by the resolver for commands which specify a VRF, such as **ping vrf vpn101 www.example.com**.
- The DNS view user1 associated with VRF vpn102 is limited to handling DNS queries from VRF vpn102 only. This view will only be used if specified inside a DNS view list that is configured for use by the DNS server globally or for a specific interface.

The two DNS views in this example can be configured with the same DNS resolving and forwarding parameters, or they can be configured with different DNS resolving and forwarding parameters.

Example: Split DNS View with Dynamic Name Server Configuration

The following example shows how to populate the list of resolving name servers for the default DNS view in the global namespace with three statically defined IP addresses. The example also shows how to configure the device to be able to dynamically acquire, through DHCP or PPP interaction on FastEthernet slot 0, port 1, name server IP addresses to add to the list of resolving name servers for that view:

```
ip dns view default
 domain lookup
 domain name-server 192.168.2.204
 domain name-server 192.168.2.205
 domain name-server 192.168.2.206
 domain name-server interface FastEthernet0/0
```

Example: Split DNS View with Statically Configured Hostname Cache Entries

The following example shows how to statically add three hostname-to-address mappings for the host www.example.com in the DNS hostname cache for the DNS view user5 that is associated with VRF vpn101:

```
clear host all *
 ip host vrf vpn101 view user5 www.example.com 192.168.2.10 192.168.2.20 192.168.2.30
 exit
show hosts vrf vpn101 view user5
```

Note It does not matter whether the VRF vpn101 has been defined. The hostname cache for this DNS view will be automatically created, and the hostname will be added to the cache.

Example: Split DNS View with Round-Robin Rotation of Hostname Cache Entries

When resolving DNS queries using a DNS view for which the hostname cache contains hostnames that are associated with multiple IP addresses, the device sends those queries to the first associated IP address in the hostname cache. By default, the other associated addresses in the hostname cache are used only in the event of host failure.

The round-robin rotation of hostname cache entries specifies that each time a hostname in the internal cache is accessed, the list of IP addresses associated with that hostname should be rotated such that the second IP address in the list becomes the first one and the first one is moved to the end of the list. For a more detailed description of round-robin functionality, see the description of the **ip domain round-robin** command in the *Cisco IOS IP Addressing Services Command Reference* .

The following example shows how to define the hostname www.example.com with three IP addresses and then enable round-robin rotation for the default DNS view associated with the global VRF. Each time that hostname is referenced internally or queried by a DNS client sending a query to the Cisco IOS DNS server on this system, the order of the IP addresses associated with the host www.example.com will be changed. Because most client applications look only at the first IP address associated with a hostname, this results in different clients using each of the different addresses and thus distributing the load among the three different IP addresses.

```
ip host view www.example.com 192.168.2.10 192.168.2.20 192.168.2.30
!
ip dns view default
 domain lookup
 domain round-robin
```

Example: Split DNS Configuration of ACLs That Can Limit DNS View Use

The following example shows how to configure one DNS name list and one standard IP ACL:

- A DNS name list is a list of hostname pattern-matching rules that can be used to restrict the use of a DNS view list member.
- A standard IP ACL is a list of IP addresses that can be used to restrict the use of a DNS view list member.

Both types of lists can be used to limit the types of DNS queries that a DNS view is allowed to handle.

```
! Define a DNS name-list
!
ip dns name-list 151 deny .*.example1.net
! (Note: The view fails this list if the query hostname matches this)
!
ip dns name-list 151 permit .*.example1.com
ip dns name-list 151 permit www.example1.org
! (Note: All other access implicitly denied)
!
! Define a standard IP ACL
!
access-list 71 deny 192.168.2.64 0.0.0.63
! (Note: The view fails this list if the query source IP matches this)
!
access-list 71 permit 192.168.2.128 0.0.0.63
! (Note: All other access implicitly denied)
```

Using this configuration example, suppose that the first member of a DNS view list is configured to use DNS name list 151 as a usage restriction. Then, if the device were to use that DNS view list to select the DNS view to use to handle a given DNS query, the view-selection steps would begin as follows:

1. If the DNS query is for a hostname that matches the string *.example1.net, the first DNS view list member is immediately rejected and the view-selection process moves on to the second member of DNS view list.
2. If the DNS query is for a hostname that matches the string *.example1.com, the first DNS view list member is selected to handle the query.
3. If the DNS query is for a hostname that matches the string www.example1.org, the first DNS view list member is selected to handle the query. Otherwise, the first DNS view list member is rejected and the view-selection process moves on to the second member of DNS view list.

Continuing to use this configuration example, suppose that this same DNS view list member is also configured to use standard IP ACL 71 as a usage restriction. Then, even if the query hostname matched DNS name list 151, the query source IP address would have to match standard IP ACL 71 before that view would be selected to handle the query. To validate this second usage restriction, the DNS view-selection steps would continue as follows:

1. If the DNS query source IP address matches 192.168.2.64, the first DNS view list member is selected to handle the query.
2. If the DNS query source IP address matches 192.168.2.128, the first DNS view list member is selected to handle the query. Otherwise, the first DNS view list member is rejected and the view-selection process moves on to the second member of the DNS view list.

Example: Split DNS View Lists Configured with Different View-use Restrictions

The following example shows how to define two DNS view lists, userlist1 and userlist2. Both view lists comprise the same three DNS views:

- DNS view user1 that is associated with the usergroup10 VRF
- DNS view user2 that is associated with the usergroup20 VRF
- DNS view user3 that is associated with the usergroup30 VRF

Both view lists contain the same DNS views, specified in the same order:

```
ip dns view-list userlist15
 view vrf usergroup100 user1 10
  restrict name-group 121
  exit
view vrf usergroup200 user2 20
  restrict name-group 122
  exit
view vrf usergroup300 user3 30
  restrict name-group 123
  exit
!
exit
ip dns view-list userlist16
 view vrf usergroup100 user1 10
  restrict name-group 121
  restrict source access-group 71
  exit
 view vrf usergroup200 user2 20
  restrict name-group 122
  restrict source access-group 72
  exit
 view vrf usergroup300 user3 30
  restrict name-group 123
  restrict source access-group 73
  exit
exit
```

The two DNS view lists differ, though, in the usage restrictions placed on their respective view list members. DNS view list userlist15 places only query hostname restrictions on its members while view list userlist16 restricts each of its members on the basis of the query hostname and the query source IP address:

- Because the members of userlist15 are restricted only based on the VRF from which the query originates, userlist15 is typical of a view list that can be used to select a DNS view for handling DNS requests from internal clients.
- Because the members of userlist16 are restricted not only by the query VRF and query hostname but also by the query source IP address, userlist16 is typical of a view list that can be used to select a DNS view for handling DNS requests from external clients.

Example: Split DNS Configuration of Default and Interface-specific View Lists

The following example shows how to configure the default DNS view list and two interface-specific view lists:

```
ip dns server view-group userlist1
!
interface FastEthernet 0/0
 ip dns view-group userlist2
 exit
!
interface FastEthernet 0/1
 ip dns view-group userlist3
 exit
```

The Cisco IOS software uses the DNS view list named userlist1 to select the DNS view to use to respond to incoming queries that arrive on device interfaces that are not configured to use a specific view list. View list userlist1 is configured as the default DNS view list for the device.

The Cisco IOS software uses the DNS view list named userlist2 to select the DNS view to use for incoming queries that arrive on port 0 of the FastEthernet card in slot 0.

The Cisco IOS software uses the DNS view list named userlist3 to select the DNS view to use for incoming queries that arrive on port 1 of the FastEthernet card in slot 0.

Additional References

Related Documents

Related Topic	Document Title
VRF-aware DNS configuration tasks: Enabling VRF-aware DNS, mapping VRF-specific hostnames to IP addresses, configuring a static entry in a VRF-specific hostname cache, and verifying the hostname cache entries in the VRF table	"VRF-Aware DNS" module
DNS configuration tasks	"Configuring DNS" module
DNS commands: complete command syntax, command mode, command history, defaults, usage guidelines, and examples	*Cisco IOS IP Addressing Services Command Reference*

Standards

Standard	Title
None	--

MIBs

MIB	MIBs Link
None	To locate and download MIBs for selected platforms, Cisco IOS releases, and feature sets, use Cisco MIB Locator found at the following URL: http://www.cisco.com/go/mibs

RFCs

RFC	Title
No new or modified RFCs are supported by this feature, and support for existing RFCs has not been modified by this feature.	--

Technical Assistance

Description	Link
The Cisco Support and Documentation website provides online resources to download documentation, software, and tools. Use these resources to install and configure the software and to troubleshoot and resolve technical issues with Cisco products and technologies. Access to most tools on the Cisco Support and Documentation website requires a Cisco.com user ID and password.	http://www.cisco.com/cisco/web/support/index.html

Feature Information for Split DNS

The following table provides release information about the feature or features described in this module. This table lists only the software release that introduced support for a given feature in a given software release train. Unless noted otherwise, subsequent releases of that software release train also support that feature.

Use Cisco Feature Navigator to find information about platform support and Cisco software image support. To access Cisco Feature Navigator, go to www.cisco.com/go/cfn. An account on Cisco.com is not required.

Table 4: Feature Information for Split DNS

Feature Name	Releases	Feature Information
Split DNS	12.4(9)T 15.4(1)T	The Split DNS feature introduces the configuration of multiple DNS databases on a device and the ability of the device to select one of these DNS server configurations based on certain characteristics of the DNS query that the device is handling. The Cisco device attempts to answer a DNS query by using the internal DNS hostname cache specified by the selected virtual DNS name server. If the DNS query cannot be answered from the information in the hostname cache, the device directs the query to specific, back-end DNS servers.

Glossary

AAA --authentication, authorization, and accounting.

ACL --access control list. A list kept by devices to control access to or from the device for a number of services (for example, to prevent packets with a certain IP address from leaving a particular interface on the device).

access control list --See ACL.

address resolution --Generally, a method for resolving differences between computer addressing schemes. Address resolution usually specifies a method for mapping network layer (Layer 3) addresses to data link layer (Layer 2) addresses.

authentication --In security, the verification of the identity of a person or a process.

bridge --Device that connects and passes packets between two network segments that use the same communications protocol. Bridges operate at the data link layer (Layer 2) of the OSI reference model. In general, a bridge filters, forwards, or floods an incoming frame based on the MAC address of that frame. See also relay.

broadcast address --A special address reserved for sending a message to all stations.

CE device --Customer edge device, an edge device in the C network, defined as a C device which attaches directly to a P device.

client --Any host requesting configuration parameters.

C network --Customer (enterprise or service provider) network.

CPE --customer premises equipment.

C device --Customer device, a device in the C network.

DDR --dial-on-demand routing. Technique whereby a device can automatically initiate and close a circuit-switched session as transmitting stations demand. The device spoofs keepalives so that end stations treat the session as active. DDR permits routing over ISDN or telephone lines using an external ISDN terminal adapter or modem.

DHCP --Dynamic Host Configuration Protocol. Provides a mechanism for allocating IP addresses dynamically so that addresses can be reused when hosts no longer need them.

DNS --Domain Name System. System used on the Internet for translating names of network nodes into addresses.

DNS name group --Association of a DNS view list member with a restriction that limits the view to handling DNS queries whose queried domain name matches a DNS name list. See also DNS source access group.

DNS name list --A named set of a domain name pattern-matching rules, with each rule specifying the type of action to be performed on a DNS query if a queried domain name matches the text string pattern.

DNS proxy --Feature that allows a device to act as a proxy for devices on the LAN by sending its own LAN address to devices that request DNS server IP addresses and forwarding DNS queries to the real DNS servers after the WAN connection is established.

DNS server view group --A DNS view list that has been configured as the default DNS view list for the device. The Cisco IOS software uses the default DNS view list to determine which DNS view to use to handle resolution of incoming DNS queries that arrive on an interface not configured with a DNS view list. See also DNS view group.

DNS source access group --Association of a DNS view list member with a restriction that limits the view to handling DNS queries whose source IP address matches a standard access control list (ACL).See also DNS name group.

DNS spoofing --Scheme used by a device to act as a proxy DNS server and “spoof” replies to any DNS queries using either the configured IP address in the **ip dns spoofing** command or the IP address of the incoming

interface for the query. This functionality is useful for devices where the interface toward the ISP is not up. Once the interface to the ISP is up, the device forwards DNS queries to the real DNS servers.

The device will respond to the DNS query with the configured IP address when queried for any hostname other than its own but will respond to the DNS query with the IP address of the incoming interface when queried for its own hostname.

The hostname used in the DNS query is defined as the exact configured hostname of the device specified by the **hostname** command, with no default domain appended.

DNS view --A named set of virtual DNS servers. Each DNS view is associated with a VRF and is configured with DNS resolver and forwarder parameters.

DNS view group --Association of a DNS view list with a device interface. The Cisco IOS software uses this view list to determine which DNS view to use to handle resolution of incoming DNS queries that arrive on that interface. See also DNS server view group.

DNS view list --A named set of DNS views that specifies the order in which the view list members should be checked and specifies usage restrictions for each view list member.

DNS view list member --A named set of DNS views that specifies the order in which the view list members should be checked and specifies usage restrictions for each view list member.

domain --On the Internet, a portion of the naming hierarchy tree that refers to general groupings of networks based on organization type or geography.

domain name --The style of identifier--a sequence of case-insensitive ASCII labels separated by dots--defined for subtrees in the Internet Domain Name System (R1034) and used in other Internet identifiers, such as hostnames, mailbox names, and URLs.

enterprise network --Large and diverse network connecting most major points in a company or other organization. Differs from a WAN in that it is privately owned and maintained.

gateway --In the IP community, an older term referring to a routing device. Today, the term router or device is used to describe nodes that perform this function, and gateway refers to a special-purpose device that performs an application-layer conversion of information from one protocol stack to another.

ISP --Internet service provider. Company that provides Internet access to other companies and individuals.

LAN --local-area network. High-speed, low-error data network covering a relatively small geographic area (up to a few thousand meters). LANs connect workstations, peripherals, terminals, and other devices in a single building or other geographically limited area. LAN standards specify cabling and signaling at the physical and data link layers of the OSI model. Ethernet, FDDI, and Token Ring are widely used LAN technologies. Compare with MAN and WAN.

MAN --metropolitan-area network. Network that spans a metropolitan area. Generally, a MAN spans a larger geographic area than a LAN, but a smaller geographic area than a WAN. Compare with LAN and WAN.

MPLS --Multiprotocol Label Switching. Switching method that forwards IP traffic using a label. This label instructs the routers and switches (or network devices) in the network where to forward the packets based on preestablished IP routing information.

multicast address --Single address that refers to multiple network devices. Synonymous with group address.

name caching --Method by which remotely discovered hostnames are stored by a device for use in future packet-forwarding decisions to allow quick access.

name resolution --Generally, the process of associating a name with a network location.

name server --Server connected to a network that resolves network names into network addresses.

namespace --Commonly distributed set of names in which all names are unique.

PE device --Provider edge device, an edge device in the P network, defined as a P device which attaches directly to a C device.

P network --MPLS-capable service provider core network. P devices perform MPLS.

P device --Provider device, a device in the P network.

relay --OSI terminology for a device that connects two or more networks or network systems. A data link layer (Layer 2) relay is a bridge; a network layer (Layer 3) relay is a router or device.

router or device --Network layer device that uses one or more metrics to determine the optimal path along which network traffic should be forwarded. Routers (or devices) forward packets from one network to another based on network layer information. Occasionally called a gateway (although this definition of gateway is becoming increasingly outdated).

server --Any host providing configuration parameters.

spoofing --Scheme used by devices to cause a host to treat an interface as if it were up and supporting a session. The device spoofs replies to keepalive messages from the host in order to convince that host that the session still exists. Spoofing is useful in routing environments, such as DDR, in which a circuit-switched link is taken down when there is no traffic to be sent across it in order to save toll charges.

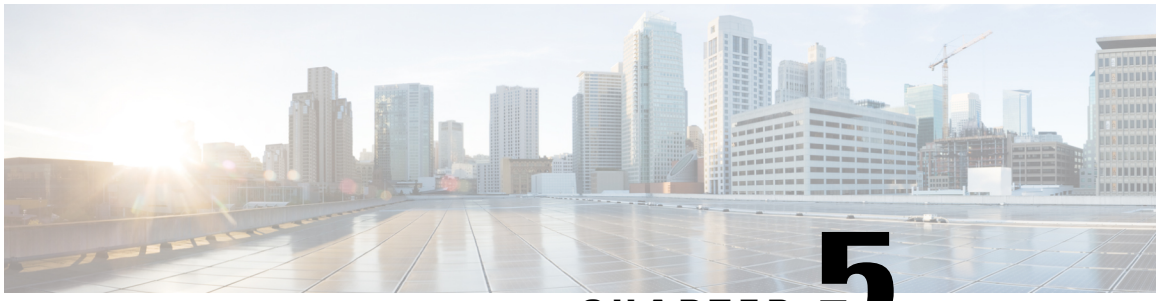
SSM --Source Specific Multicast. A datagram delivery model that best supports one-to-many applications, also known as broadcast applications. SSM is the core networking technology for the Cisco implementation of the IP Multicast Lite suite of solutions targeted for audio and video broadcast application environments.

tunnel --Secure communication path between two peers, such as two devices.

VPN --Virtual Private Network. Framework that consists of multiple peers transmitting private data securely to one another over an otherwise public infrastructure. A VPN protects inbound and outbound network traffic by using protocols that tunnel and encrypt all data at the IP level. This framework permits networks to extend beyond their local topology, while remote users are provided with the appearance and functionality of a direct network connection. Enables IP traffic to travel securely over a public TCP/IP network by encrypting all traffic from one network to another. A VPN uses “tunneling” to encrypt all information at the IP level.

VRF --VPN routing and forwarding instance. A VRF consists of an IP routing table, a derived forwarding table, a set of interfaces that use the forwarding table, and a set of rules and routing protocols that determine what goes into the forwarding table. In general, a VRF includes the routing information that defines a customer VPN site that is attached to a PE device. Each VPN instantiated on the PE device has its own VRF.

WAN --wide-area network. Data communications network that serves users across a broad geographic area and often uses transmission devices provided by common carriers. Frame Relay, SMDS, and X.25 are examples of WANs. Compare with LAN and MAN.



CHAPTER 5

Service Discovery Gateway

The Service Discovery Gateway feature enables multicast Domain Name System (mDNS) to operate across Layer 3 (L3) boundaries. An mDNS gateway will be able to provide transport for service discovery across L3 boundaries by filtering, caching and extending services from one subnet to another. Prior to implementation of this feature, mDNS was limited in scope to within a subnet due to the use of link-local scoped multicast addresses. This feature enhances Bring Your Own Device (BYOD).

Caution Extension of services should be done with proper care. Generally, only specific services should be extended. Service names should be unique in the network to avoid duplicate name conflicts.

See Feature Information for Service Discovery Gateway section to check feature availability for your platform release version.

- Information About Service Discovery Gateway, page 93
- How to Configure Service Discovery Gateway, page 99
- Verifying and troubleshooting Service Discovery Gateway, page 106
- Configuration Examples for Service Discovery Gateway, page 108
- Additional References for Service Discovery Gateway , page 111
- Feature Information for Service Discovery Gateway , page 112

Information About Service Discovery Gateway

Service Announcement Redistribution and Service Extension

Redistribution of announcements is the actual forwarding of announcements and query responses while service extension is the capability of proxying services between subnets. The actual replication of the service announcement can help to speed up the visibility of newly announced services and also a service's withdrawal if a service or device is turned off.

Note Extension of services such as printers or Apple TV works fine without actual replication of service announcements. The Service Discovery Gateway will cache announcements, queries and their responses in the cache. If another device queries for a service, the Service Discovery Gateway will be able to provide an answer from its cache.

Enable the **redistribution mdns-sd** command only on a per-interface basis, and only if it is actually required. You must ensure that there are no loops in the network topology corresponding to the interface for which service announcement redistribution is being enabled. A loop can lead to a broadcast storm.

Redistribution of service announcement information cannot be done globally. You can enable redistribution of service information only at the interface level.

Extending Services Across Subnets—An Overview

You need to enable a multicast Domain Name System (mDNS) gateway to extend services across subnet boundaries. You can enable an mDNS gateway for a device or for an interface. You must enable routing of services for the device before enabling it at the interface level. After the mDNS gateway is enabled on a device or interface, you can extend services across subnet boundaries.

To extend services across subnets, you must do the following:

1. Set Filter Options to Extend Services Across Subnets—You can allow services such as printer services to be accessed across subnets. If printer x is available on interface 1, users on interface 2 can use printer x without configuring the printer on their local systems.
2. Extend Services Across Subnets—The filter created in Step 1 should be applied on the interfaces 1 and 2. Only then can users on other interfaces access the printer service.

For the sample scenario where a printer service is accessible by clients on other interfaces, you must apply these filters:

- On the interface where the printer service is available (IN filter) —You want to allow the printer service *into* the mDNS cache, so that it can be accessed by users on other subnets.
- On the interface where the printer service is available (OUT filter)—Since clients on other interfaces will access the service (printer x, for example), you should allow queries coming from the device (OUT filter, from the device's point of view).
- On each interface where clients reside (IN filter)—For clients on other interfaces (subnets) wanting to access the printer service, you must allow queries from users into the mDNS cache (IN filter).

Remember Applying the IN filter means that you are allowing the printer service into the device mDNS cache, and other interfaces can access it. Applying the OUT filter means that you are allowing the queries out of the cache so that queries from clients on other interfaces can reach the printer interface. On other client-facing interfaces, the IN filter is applied to allow queries in.

Note

- Filters can be applied at the global level and at the interface level. Filters applied at the interface level takes precedence over the filters applied at the global level.
- The term 'service discovery information' refers to services (printer services, etc), queries (queries for printer services, etc, from one interface to the other), announcements (printer service is removed, etc), and service-instances (a specific service—printer x, Apple TV 3, etc) that you want to extend across subnets.

Set Filter Options to Extend Services Across Subnets

You can set filter options to allow services such as printer services into or out of a device or interface. You can also permit or prohibit queries, announcements, services learnt from an interface, specific service–instances, and locations. Use the **service-list mdns-sd** command to create a service-list and set filter options.

You need to create a service-list and use filter options within it. While creating a service-list, use one of the following options:

- The **permit** option permits specific services, announcements and service-instances across subnets.
- The **deny** option restricts services, announcements and service-instances from being transported across subnets.
- The **query** option is provided to browse services. For example, if you want to browse printer services periodically, then you can create a service-list with the **query** option, and add the printer service to the query. When you set a period for the query, the service entries are refreshed in the cache memory.

You must mention a sequence number when using the **permit** or **deny** option. The filtering is done sequentially, in the ascending order. The same service-list can be associated with multiple sequence numbers. Within a sequence, match statements (commands) must be used to specify what needs to be filtered. Generally, match statements are used to filter queries (for example, queries from clients to find printer and fax services), announcements (new service is added, and so on), specific service–instances, types of service such as printer services (so that the service is allowed into the cache for use), services available for a specific interface (printers and Apple TVs associated with a VLAN), and locations.

Note A service-list by itself does not contain any services. You must specify a service type in the match statement when setting filter options to allow or prohibit services. (For example, '_ipp._tcp' is the service type for an IPP printing service running over TCP).

Sample scenario - Consider a device is in a client segment. The goal is to allow the following on the device:

- All queries from clients to the device.
- Printer services to clients on other subnets.

The following example explains how to achieve the goal:

```
!
service-list mdns-sd mixed permit 10
  match message-type query
```

```
!
service-list mdns-sd mixed permit 20
  match message-type announcement
  match service-type _ipps._tcp.local
!
```

In the above example, a service-list called 'mixed' is created and the **permit** option is used twice—to filter queries and to filter printer services and announcements. The filtering is done in the sequence given below:

- Sequence 10 - A match statement is used to filter queries.
- Sequence 20 - Match statements are used to filter announcements and printer services.

The match statement in Sequence 10 sets a filter for queries on the device, but does not specify that queries be allowed *into* the device. To allow queries from clients, the filter needs to be applied on the interface in the IN direction. The example is displayed in the Extend Services Across Subnets section.

Similarly, the match statements in Sequence 20 sets a filter for announcements and printer services on the device, but does not specify that they be allowed *into* the device. To allow announcements and printer services into the device, the filter needs to be applied on the required interfaces in the IN direction. The example is displayed in the Extend Services Across Subnets section.

If neither the **permit** option nor the **deny** option is used, the default action is to disallow services from being transported to other subnets.

Browsing services periodically—Service-lists of the type **query** can be used to browse services. Such queries are called active queries. Active queries periodically send out requests for the services specified within the query on all interfaces. As services have a specific Time to Live (TTL) duration, active queries can help to keep services fresh in the cache memory.

In the following example, a service-list named 'active-query' is created and the service-list is of the type **query**. Services such as printer services are specified within the query, and these are the services that we want to extend. Typically, these services would match the services that have been configured as 'permitted' services in the IN filter.

```
!
service-list mdns-sd active-query query
  service-type _universal._sub._ipp._tcp
  service-type _ipp._tcp.local
  service-type _ipps._tcp.local
  service-type _raop._tcp.local
!
```

The purpose of an active query and a query associated with a match statement is different. When you enable an active query, services are browsed periodically. A query is used in a match statement to permit or prohibit queries (not active queries) on the interface.

Note

- Service-list creation can only be used globally and cannot be used at the interface level.
- You can create a new service-instance of a specific service-type using the **service-instance mdns-sd** command.
- A service end–point (such as a printer, fax, and so on) sends unsolicited announcements when a service starts up. After that, it sends unsolicited announcements whenever a network change event occurs (such as, an interface coming up or going down, and so on). The device always responds to queries.

Remember Filtering only sets filter options and specifies that certain services need to be filtered. You must *apply* the filters on an interface for the services, queries, or announcements to actually be permitted or prohibited on the interface. To know about applying filters and the other available service discovery configuration options, refer the Extend Services Across Subnets section.

Extend Services Across Subnets

You must have set filter options for the device before extending services across subnets. If you have set filter options for specific services and other service discovery information to be allowed, prohibited or queried periodically, you can apply the filters for an interface.

Before applying filters, note the following:

- You must enable multicast Domain Name System (mDNS) on a device to apply filter options. You can enable mDNS using the command **service-routing mdns-sd**
- Since you might want to allow services into the device or prohibit services from being learnt on an interface, you must apply the filter in the needed direction. The options **IN** and **OUT** perform the desired actions on the interface.
- Typically, a service-policy is applied on an interface. Global service-policies are optional and affect all L3 interfaces.

Sample scenario - A device is in a client segment and the goal is to allow the following between the device interfaces:

- All queries from clients to the device.
- Printer services.

A note about filter options - Filter options have been set for the above scenario by creating a service-list called 'mixed' and adding filter options to it. (see Set Filter Options to Extend Services Across Subnets for more details). The following example explains how to apply the filters:

```
!
interface Ethernet0/0
 description *** (wireless) Clients here plus some printers
 ip address 172.16.33.7 255.255.255.0
 service-routing mdns-sd
  service-policy mixed IN
!
interface Ethernet0/3
 description *** (wireless) Clients here plus some printers
 ip address 172.16.57.1 255.255.255.0
 service-routing mdns-sd
  service-policy mixed IN
!
```

In the above example, service-routing is enabled on the interface and the filter options in the service-policy 'mixed' are applied in the **IN** direction. In other words, all queries and printer services will be allowed into the device, from the interfaces Ethernet 0/0 and Ethernet 0/3.

Sample scenario for browsing specific services - A service-list of the type **query** (called active query) has been created. It contains services that we want to browse periodically, such as printer services (see Set Filter

Options to Extend Services Across Subnets for more details about creating an active query). To enable browsing of the services in the query, you must apply the active query for the device.

```
!
service-routing mdns-sd
 service-policy-query active-query 900
!
```

In the above example, the period is set to 900 seconds. The services within the active query are queried on all interfaces of the device after an interval of 900 seconds.

Note

- You can enable browsing of services for specific interfaces. If browsing of services is enabled globally, you can disable browsing of services on specific interfaces.
- Services are browsed specific to a device or interface by the mDNS process. So, the IN or OUT option is not relevant for browsing of services.

You can use the following options after enabling mDNS on a device or interface.

Purpose	**Use this Command** **Note** The complete syntax is provided in the corresponding task.	**Global and Interface Configuration Options**
For a service-list, apply a filter to allow or prohibit services.	**service-policy**	Global and interface levels.
Set some part of the system memory for cache.	**cache-memory-max**	Global level.
Configure an active query and the query period so that specified services are queried periodically.	**service-policy-query**	Global level.
Designate a specific device or interface in a domain for routing mDNS announcement and query information.	**designated-gateway**	Global and interface levels.
Access services in the proximity of the device. **Note** Service policy proximity filtering functionality is only available on wireless devices and their interfaces.	**service-policy-proximity**	Global and interface levels.
Configure service-type enumeration period for the device.	**service-type-enumeration period**	Global level.

Specify an alternate source interface for outgoing mDNS packets on a device.	**source-interface**	Global level.
Configure the maximum rate limit of incoming mDNS packets for a device.	**rate-limit**	Global level.
Speed up visibility of newly announced services and withdrawal of services when a service or device is turned off.	**redistribute**	Interface level.

How to Configure Service Discovery Gateway

Setting Filter Options for Service Discovery

Before You Begin

Ensure that you permit a query or announcement when you set filter options. If you do not use a **permit** option and only use **deny** options, you will not be able to apply the filter.

SUMMARY STEPS

1. **enable**
2. **configure terminal**
3. **service-list mdns-sd** *service-list-name* {**deny** *sequence-number* | **permit** *sequence-number* | **query**}
4. **match message-type** {**announcement** | **any** | **query**}
5. **match service-instance** {*instance-name* | **any** | **query**}
6. **match service-type** *mDNS-service-type-string*
7. **match location civic** *civic-location-name*
8. **exit**

DETAILED STEPS

	Command or Action	Purpose
Step 1	**enable** **Example:** `Device> enable`	Enables privileged EXEC mode. • Enter your password if prompted.

	Command or Action	Purpose
Step 2	**configure terminal** **Example:** `Device# configure terminal`	Enters global configuration mode.

Step 3	**service-list mdns-sd** *service-list-name* {**deny** *sequence-number* | **permit** *sequence-number* | **query**}
	
	Example:
	
	`Device(config)# service-list mdns-sd sl1 permit 3`
	
	`Or`
	
	`Device(config)# service-list mdns-sd sl4 query`
		Enters mdns service discovery service-list mode.
		
		• Creates a service-list and applies a filter on the service-list according to the **permit** or **deny** option applied to the sequence number.
		
		Or
		
		• Creates a service-list and associates a query for the service-list name if the **query** option is used.
		
		Remember When you set filter options, ensure that you permit a query or announcement for a service-list. If you do not use a **permit** option and only use **deny** options, you will not be able to apply the filter.

Step 4	**match message-type** {**announcement** | **any** | **query**} **Example:** `Device(config-mdns-sd-sl)# match message-type announcement`	Configures parameters for a service-list based on a service announcement or query. **Note** You cannot use the **match** command if you have used the **query** option. The **match** command can be used only for the **permit** or **deny** option.
Step 5	**match service-instance** {*instance-name* | **any** | **query**} **Example:** `Device(config-mdns-sd-sl)# match service-instance printer-3`	Configures parameters for a service-list based on a service-instance or query.
Step 6	**match service-type** *mDNS-service-type-string* **Example:** `Device(config-mdns-sd-sl)# match service-type _ipp._tcp.local`	Configures parameters for a service-list based on a service-type.
Step 7	**match location civic** *civic-location-name* **Example:** `Device(config-mdns-sd-sl)# match location civic location3`	Configures parameters for a service-list based on a civic location.

	Command or Action	Purpose
Step 8	**exit** **Example:** `Device(config-mdns-sd-sl)# exit`	Exits mdns service discovery service-list mode, and returns to global configuration mode.

What to Do Next

Apply filters on an interface for the services, queries, or announcements to actually be permitted or prohibited on the interface.

Applying Service Discovery Filters and Configuring Service Discovery Parameters

After enabling multicast Domain Name System (mDNS) gateway for a device, you can apply filters (IN-bound filtering or OUT-bound filtering) and active queries by using **service-policy** and **service-policy-query** commands, respectively.

Note Steps 5 to 11 are mDNS Service Discovery configuration options. The steps are optional and not meant to be used in any specific order.

Before You Begin

You must set filter options for the device before applying filters.

SUMMARY STEPS

1. **enable**
2. **configure terminal**
3. **service-routing mdns-sd**
4. **service-policy** *service-policy-name* {**IN** | **OUT**}
5. **cache-memory-max** *cache-config-percentage*
6. **service-policy-query** *service-list-name query-period*
7. **designated-gateway enable** [**ttl** *duration*]
8. **service-policy-proximity** *service-list-name* [**limit** *number-of-services*]
9. **service-type-enumeration period** *period-value*
10. **source-interface** *type number*
11. **rate-limit in** *maximum-rate-limit*
12. **exit**

DETAILED STEPS

	Command or Action	Purpose
Step 1	**enable** **Example:** `Device> enable`	Enables privileged EXEC mode. • Enter your password if prompted.
Step 2	**configure terminal** **Example:** `Device# configure terminal`	Enters global configuration mode.
Step 3	**service-routing mdns-sd** **Example:** `Device(config)# service-routing mdns-sd`	Enables mDNS gateway functionality for a device and enters multicast DNS configuration (config-mdns) mode.
Step 4	**service-policy** *service-policy-name* {**IN** | **OUT**} **Example:** `Device(config-mdns)# service-policy sl1 IN`	For a service-list, applies a filter on incoming service discovery information (IN-bound filtering) or outgoing service discovery information (OUT-bound filtering). **Note** Global service-policies are optional and effect all L3 interfaces. Typically, a service-policy is applied on an interface.
Step 5	**cache-memory-max** *cache-config-percentage* **Example:** `Device(config-mdns)# cache-memory-max 20`	Sets some part of the system memory (in percentage) for cache. **Note** By default, 10% of the system memory is set aside for cache. You can override the default value by using this command.
Step 6	**service-policy-query** *service-list-name query-period* **Example:** `Device(config-mdns)# service-policy-query sl4 100`	Creates an active query and configures the service-list-query period.
Step 7	**designated-gateway enable** [**ttl** *duration*] **Example:** `Device(config-mdns)# designated-gateway enable`	Designates the device to route mDNS announcement and query information for the domain.
Step 8	**service-policy-proximity** *service-list-name* [**limit** *number-of-services*]	Configures service policy proximity filtering on the device. • Service policy proximity filtering is only available for wireless clients and is based on Radio Resource

	Command or Action	Purpose
	Example: `Device(config-mdns)# service-policy-proximity sl1 limit 10`	Management (RRM). Wired clients and services are not affected by the limit. • The default value for the maximum number of services that can be returned is 50.
Step 9	**service-type-enumeration period** *period-value* **Example:** `Device(config-mdns)# service-type-enumeration period 45`	Configures service-type enumeration period for the device.
Step 10	**source-interface** *type number* **Example:** `Device(config-mdns)# source-interface ethernet 0/1`	Specifies an alternate source interface for outgoing mDNS packets on a device.
Step 11	**rate-limit in** *maximum-rate-limit* **Example:** `Device(config-mdns)# rate-limit in 80`	Configures the maximum rate limit of incoming mDNS packets for a device.
Step 12	**exit** **Example:** `Device(config-mdns)# exit`	Exits multicast DNS configuration mode, and returns to global configuration mode.

Applying Service Discovery Filters for an Interface

SUMMARY STEPS

1. **enable**
2. **configure terminal**
3. **interface** *type number*
4. **service-routing mdns-sd**
5. **service-policy** *service-policy-name* {**IN** | **OUT**}
6. **redistribute mdns-sd**]
7. **exit**

DETAILED STEPS

	Command or Action	Purpose
Step 1	**enable** **Example:** `Device> enable`	Enables privileged EXEC mode. • Enter your password if prompted.
Step 2	**configure terminal** **Example:** `Device# configure terminal`	Enters global configuration mode.
Step 3	**interface** *type number* **Example:** `Device(config)# interface ethernet 0/1`	Enters Interface multicast DNS configuration mode, and enables interface configuration.
Step 4	**service-routing mdns-sd** **Example:** `Device(config-if)# service-routing mdns-sd`	Enables mDNS gateway functionality for an interface and enters multicast DNS configuration (config-mdns) mode.
Step 5	**service-policy** *service-policy-name* {**IN** | **OUT**} **Example:** `Device(config-if-mdns-sd)# service-policy sl1 IN`	For a service-list, applies a filter on incoming service discovery information (IN-bound filtering) or outgoing service discovery information (OUT-bound filtering). **Remember** When you set filter options, ensure that you permit a query or announcement for a service-list. If you have not permitted a service, query, or announcement while setting filter options, then you will see this warning when you apply the filter: **Warning**: Please enable explicit service-list rule with the permit action to allow queries and responses.
Step 6	**redistribute mdns-sd**] **Example:** `Device(config-if-mdns-sd)# redistribute mdns-sd`	Speeds up visibility of newly announced services when a service or device is turned off.
Step 7	**exit** **Example:** `Device(config-if-mdns-sd)# exit`	Exits Interface multicast DNS configuration mode, and returns to interface configuration mode.

Creating a Service Instance

SUMMARY STEPS

1. **enable**
2. **configure terminal**
3. **service-instance mdns-sd service** *instance-name* **regtype** *service-type* **domain** *name*
4. {**ipv4addr** | **ipv6addr**} *IP-address*
5. **port** *number*
6. **target-hostname** *host-name*
7. **txt** *text-record-name*
8. **priority** *value*
9. **weight** *value*
10. **exit**

DETAILED STEPS

	Command or Action	Purpose
Step 1	**enable** **Example:** `Device> enable`	Enables privileged EXEC mode. • Enter your password if prompted.
Step 2	**configure terminal** **Example:** `Device# configure terminal`	Enters global configuration mode.
Step 3	**service-instance mdns-sd service** *instance-name* **regtype** *service-type* **domain** *name* **Example:** `Device(config)# service-instance mdns-sd service printer-3 regtype _ipp._tcp.local domain tcp4`	Creates a service-instance of a specific service type and enters multicast Domain Name System (mDNS) service discovery service-instance (config-mdns-sd-si) mode. **Note** In this mode, you can configure various parameters for the service-instance. The subsequent steps show how to configure service-instance parameters.
Step 4	{**ipv4addr**	**ipv6addr**} *IP-address* **Example:** `Device(config-mdns-sd-si)# ipv4addr 209.165.200.230 255.255.255.0`	Specifies the IPv4 or IPv6 address of the port on which the service is available.

	Command or Action	Purpose
Step 5	**port** *number* **Example:** `Device(config-mdns-sd-si)# port 9100`	Specifies the port on which the service is available.
Step 6	**target-hostname** *host-name* **Example:** `Device(config-mdns-sd-si)# target-hostname fqdn-of-printer.example.com.`	Specifies the fully qualified domain name (FQDN) of the target host.
Step 7	**txt** *text-record-name* **Example:** `Device(config-mdns-sd-si)# txt _ipp._tcp.local=printer3`	Specifies the text record associated with the service instance. **Note** A TXT record is a type of DNS record that provides text information to sources outside your domain. Specify the text record in the format 'service-type=service-name'. To specify multiple records, use a semicolon (;) as a separator.
Step 8	**priority** *value* **Example:** `Device(config-mdns-sd-si)# priority 3`	(Optional) Specifies the priority value for the service-instance. The default priority value is zero.
Step 9	**weight** *value* **Example:** `Device(config-mdns-sd-si)# weight 20`	(Optional) Specifies the weight value for the service-instance. The default weight value is zero.
Step 10	**exit** **Example:** `Device(config-mdns-sd-si)# exit`	Exits multicast Domain Name System (mDNS) service discovery service-instance (config-mdns-sd-si) mode and enters global configuration mode.

Verifying and troubleshooting Service Discovery Gateway

Note The show and debug commands mentioned below are not in any specific order.

SUMMARY STEPS

1. **show mdns requests** [**detail** | [**type** *record-type*] [**name** *record-name*]]
2. **show mdns cache** [**interface** *type number* [**detail**] | [**name** *record-name*] [**type** *record-type*] [**detail**]]
3. **show mdns statistics** {**all** | **interface** *type number* | **service-list** *list-name* | [**cache** | **service-policy**] {**all** | **interface** *type number*} | **services orderby providers**}
4. **show mdns service-types** [**all** | **interface** *type number*]
5. **debug mdns** {**all** | **error** | **event** | **packet** | **verbose**}

DETAILED STEPS

Step 1 **show mdns requests** [**detail** | [**type** *record-type*] [**name** *record-name*]]

Example:

```
Device# show mdns requests detail

MDNS Outstanding Requests
=========================================================
Request name  :   _ipp._tcp.local
Request type  :   PTR
Request class :   IN
```

This command displays information for outstanding multicast Domain Name System (mDNS) requests, including record name and record type information.

Step 2 **show mdns cache** [**interface** *type number* [**detail**] | [**name** *record-name*] [**type** *record-type*] [**detail**]]

Example:

Note You can use the **detail** keyword for a specific interface, record or type. You cannot use it independently with the **show mdns cache** command.

```
Device# show mdns cache

mDNS CACHE
==========================================================================================
[<NAME>]                                                [<TYPE>][<CLASS>] [<TTL>/Remaining] [Accessed]
 [If-index] [<RR Record Data>]


_services._dns-sd._udp.local                              PTR     IN      4500/4496              0
     3       _ipp._tcp.local

_ipp._tcp.local                                           PTR     IN      4500/4496              1
     3       printer1._ipp._tcp.local

printer1._ipp._tcp.local                                  SRV     IN      120/116         1      3
     0       0       5678    much-WS.local

printer1._ipp._tcp.local                                  TXT     IN      4500/4496              1
     3       (1)''

music-WS.local                                          A       IN      120/116         1      3
  192.168.183.1
```

This command displays mDNS cache information.

Step 3 **show mdns statistics** {**all** | **interface** *type number* | **service-list** *list-name* | [**cache** | **service-policy**] {**all** | **interface** *type number*} | **services orderby providers**}

Example:

```
Device# show mdns statistics all

mDNS Statistics
 mDNS packets sent       : 0
 mDNS packets received   : 31
 mDNS packets dropped    : 8
 mDNS cache memory in use: 64264(bytes)
```

This command displays mDNS statistics.

Step 4 **show mdns service-types** [**all** | **interface** *type number*]

Example:

```
Device# show mdns service-types

mDNS SERVICES
================================
[<NAME>]          [<TTL>/Remaining] [If-name]
_ipp._tcp.local     4500/4496
```

This command displays mDNS statistics.

Step 5 **debug mdns** {**all** | **error** | **event** | **packet** | **verbose**}

Example:

```
Device# debug mdns all
```

This command enables all mDNS debugging flows.

Configuration Examples for Service Discovery Gateway

Example: Setting Filter Options for Service Discovery

The following example shows creation of a service-list sl1. The permit option is being applied on sequence number 3 and all services with message-type announcement are filtered and available for transport across various subnets associated with the device.

```
Device> enable
Device# configure terminal
Device(config)# service-list mdns-sd sl1 permit 3
Device(config-mdns-sd-sl)# match message-type announcement
Device(config-mdns-sd-sl)# exit
```

Example: Applying Service Discovery Filters and Configuring Service Discovery Parameters

```
Device> enable
Device# configure terminal
```

```
Device(config)# service-routing mdns-sd
Device(config-mdns)# service-policy serv-pol1 IN
Device(config-mdns)# cache-memory-max 20
Device(config-mdns)# service-policy-query sl-query1 100
Device(config-mdns)# designated-gateway enable
Device(config-mdns)# rate-limit in 80
Device(config-mdns)# exit
```

Example: Applying Service Discovery Filters for an Interface

```
Device> enable
Device# configure terminal
Device(config)# interface ethernet 0/1
Device(config-if)# service-routing mdns-sd
Device(config-if-mdns-sd)# service-policy sl1 IN
Device(config-if-mdns-sd)# redistribute mdns-sd
Device(config-if-mdns-sd)# exit
```

Example: Setting Multiple Service Discovery Filter Options

The following example shows creation of filters using service-lists mixed, permit-most, permit-all, and deny-all. Then, the filters are applied at various interfaces, as required.

```
!
service-list mdns-sd mixed permit 10
 match message-type query
!
service-list mdns-sd mixed permit 20
 match message-type announcement
 match service-type _ipps._tcp.local
!
service-list mdns-sd mixed permit 30
 match message-type announcement
 match service-type _ipp._tcp.local
 match service-type _universal._sub._ipp._tcp
!
service-list mdns-sd mixed permit 40
 match message-type announcement
!
service-list mdns-sd mixed deny 50
!
service-list mdns-sd permit-most deny 10
 match service-type _sleep-proxy._udp.local
!
service-list mdns-sd permit-most permit 20
!
service-list mdns-sd permit-all permit 10
!
service-list mdns-sd deny-all permit 10
 match message-type query
!
service-list mdns-sd deny-all deny 20
!
service-list mdns-sd active-query query
 service-type _universal._sub._ipp._tcp.local
 service-type _ipp._tcp.local
 service-type _ipps._tcp.local
 service-type _raop._tcp.local
!
service-routing mdns-sd
 service-policy-query active-query 900
!
```

```
!
interface Ethernet0/0
 description *** (wireless) Clients here plus some printers or aTVs
 ip address 172.16.33.7 255.255.255.0
 service-routing mdns-sd
  service-policy mixed IN
  service-policy permit-all OUT
!
interface Ethernet0/1
 description *** AppleTVs, Print Servers here
 ip address 172.16.57.1 255.255.255.0
 service-routing mdns-sd
  service-policy permit-most IN
  service-policy permit-all OUT
!
interface Ethernet0/2
 description *** Clients only, we don't want to learn anything here
 ip address 172.16.58.1 255.255.255.0
 service-routing mdns-sd
  service-policy deny-all IN
  service-policy permit-all OUT
!
interface Ethernet0/3
 no ip address
 shutdown
!
```

In the above example, the service-lists are:

- permit-all - As the name suggests, this service-list permits all resource records, and should be used with care. This is typically applied in the OUT direction; allows the cache to respond to all requests regardless of query content or query type.
- permit-most - This allows anything in, except for sleep-proxy services. This is because extending sleep-proxy services causes an issue with devices that register with a sleep proxy across the Service Discovery Gateway. Due to split horizon, the real (sleeping) device won't be able to re-register its services when waking up again when its pointer (PTR) record is pointing to the sleep-proxy.
- deny-all - This prevents the cache from learning anything. Again incoming on a segment where only clients live. As a result, clients will be able to query for services from the cache (hence the permit 10 match query), but there is no need to learn anything from the clients.
- mixed - This is created to be used in client segments. In addition to clients (such as iPads, PCs, and so on), the occasional printer or a TV will also connect. The purpose here is to learn about those specific services but not about services the clients provide. The filter applied is IN. As a result, the following actions are applicable:
  - Allow every query IN.
  - Allow specific services in (such as printer services [IPP]).
  - Deny everything else.

In addition, to keep the service PTRs fresh in the cache an active query is configured. The active query queries for those services that we want to extend. Typically, this would match the services that have been configured as 'permitted' services in the IN filter. The value is set to 900 seconds. The duration is enough to refresh the PTRs as they typically have a TTL of 4500 seconds.

Example: Creating a Service Instance

```
Device> enable
Device# configure terminal
Device(config)# service-instance mdns-sd service printer-3 regtype _ipp._tcp.local domain
tcp4
Device(config-mdns-sd-si)# ipv4addr 209.165.200.230 255.255.255.0
Device(config-mdns-sd-si)# port 9100
Device(config-mdns-sd-si)# target-hostname fqdn-of-printer.example.com.
Device(config-mdns-sd-si)# txt _ipp._tcp.local=printer3
Device(config-mdns-sd-si)# priority 3
Device(config-mdns-sd-si)# weight 20
Device(config-mdns-sd-si)# exit
```

Note When you create a service-instance, a text record is created even if you do not configure service-instance parameters.

Additional References for Service Discovery Gateway

Related Documents

Related Topic	Document Title
Master Command List	Cisco IOS Master Command List
IP Addressing Services Command Reference	Cisco IOS IP Addressing Services Command Reference
Configuring DNS	IP Addressing: DNS Configuration Guide
DNS conceptual information	“Information About DNS” section in IP Addressing: DNS Configuration Guide

Standards and RFCs

Standard/RFC	Title
RFC 6762	Multicast DNS
RFC 6763	DNS-Based Service Discovery
Multicast DNS Internet-Draft	Multicast DNS Internet draft

MIBs

MIB	MIBs Link
No new or modified MIBs are supported by this feature, and support for existing MIBs has not been modified by this feature.	To locate and download MIBs for selected platforms, Cisco IOS releases, and feature sets, use Cisco MIB Locator found at the following URL: http://www.cisco.com/go/mibs

Technical Assistance

Description	Link
The Cisco Support website provides extensive online resources, including documentation and tools for troubleshooting and resolving technical issues with Cisco products and technologies. To receive security and technical information about your products, you can subscribe to various services, such as the Product Alert Tool (accessed from Field Notices), the Cisco Technical Services Newsletter, and Really Simple Syndication (RSS) Feeds. Access to most tools on the Cisco Support website requires a Cisco.com user ID and password.	http://www.cisco.com/support

Feature Information for Service Discovery Gateway

The following table provides release information about the feature or features described in this module. This table lists only the software release that introduced support for a given feature in a given software release train. Unless noted otherwise, subsequent releases of that software release train also support that feature.

Use Cisco Feature Navigator to find information about platform support and Cisco software image support. To access Cisco Feature Navigator, go to www.cisco.com/go/cfn. An account on Cisco.com is not required.

Table 5: Feature Information for Service Discovery Gateway

Feature Name	Releases	Feature Information
Service Discovery Gateway	15.1(2)SY	The Service Discovery Gateway feature enables multicast Domain Name System (mDNS) to operate across L3 boundaries (different subnets). The following commands were introduced or modified: **cache-memory-max, clear mdns cache, clear mdns statistics, debug mdns, match message-type, match service-instance, match service-type, redistribute mdns-sd, service-list mdns-sd, service-policy, service-policy-query, service-routing mdns-sd, show mdns cache, show mdns requests, show mdns statistics**
Service Discovery Gateway—Phase 2	15.2(1)SY	The Service Discovery Gateway feature was enhanced with additional filter and configuration options. The following commands were introduced or modified: **clear mdns cache, clear mdns service-types, clear mdns statistics, designated-gateway, match location, rate-limit, service-instance mdns-sd, service-policy-proximity, service-routing mdns-sd, service-type-enumeration, show mdns cache, show mdns statistics, source-interface**