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

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

CHAPTER 1

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.

Note Service Discovery Gateway features have been incrementally added, in phases. For example, features added for phase three are:

- De-congestion of incoming mDNS traffic using rate limiting mechanism.
- Redistribution of messages when services are withdrawn, to improve mDNS cache efficiency and to avoid message loops.
- A filter criterion (a **match** command) for services available and learnt on a specific interface.
- Enabling and disabling of periodic browsing of services on specific interfaces; previously, services could be browsed on all interfaces of a device but not specific interfaces.

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

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.

The withdraw-only option

When you use this option, a service withdrawal announcement is sent to other devices when a service is removed, and the service is removed from the device's mDNS cache too. When you use the **withdraw-only** option, redistribution is only enabled for service withdrawal and not for the service announcements. For example, if the service withdrawal announcement option for a printer service is enabled, announcements about the printer service availability are not sent on other subnets. However, if the printer service is removed, withdrawal announcements will be sent to other device interfaces where the printer service is learnt.

If you do not use this option, the service will still be seen as available on other Service Discovery Gateway enabled device interfaces which have queried for this service earlier and stored in the mDNS cache. As a result, users connected to other SDG-enabled devices will not see the withdrawn service as available.

Note

- In scenarios when an active query is enabled, message loops are created because the original gateway relearns the service from another gateway even after the service has been withdrawn. To avoid such situations, this option was introduced.
- The **withdraw-only** option is not available for wireless devices.

The **withdraw-only** option is availble on specific release versions. See Feature Information for Service Discovery Gateway section to check feature availability for your platform release version.

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.

De-congestion of Incoming mDNS Traffic Using the Rate Limiting Mechanism

Incoming mDNS packets are subjected to rate limiting by the mDNS process. The mDNS process maintains statistics, including the rate of incoming and out going messages. You can configure a rate limit value in the range 1-100 packets per second (p/s).

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.

Note To filter services learnt for an interface, use the **match learnt-interface** command. This command was included from a specific release onwards. See the Feature Information for Service Discovery Gateway section to check its availability for your platform release version.

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.

Customizing browsing of services

- If you enable browsing of printer services globally, printer services will be periodically queried on all interfaces. If you do not want one or more interfaces to browse for the printer service, use the **disable** option on the interface. Printer services will not be browsed on the interface. The **disable** option can only be used for specific interfaces.
- You can enable browsing of services for specific interfaces too. For example, you can enable periodic browsing for printer services only on interfaces where you have printers connected.
- When you create a service-list for browsing services, ensure that you plan what services you want to include in a service-list. When the **disable** option is used on an interface, browsing of all services within the service-list will be discontinued on the interface.

Note The **disable** option and the option to enable browsing of services for specific interfaces have been included from a specific release onwards. See the Feature Information for Service Discovery Gateway section to check feature availability for your platform release version.

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 and interface levels.

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. **match learnt-interface** *interface number*
9. **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-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.

	Command or Action	Purpose

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.
Step 8	**match learnt-interface** *interface number* **Example:**	Filters services that are available on an interface and associates the filtered data to a specific service-list. **Note** For example, if a printer service on VLAN 1 is filtered under a service-list using this command, and if you apply the filters on VLAN 2, the printer will be available for use by clients in VLAN 2.
Step 9	**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.

	Command or Action	Purpose

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*] **Example:** `Device(config-mdns)# service-policy-proximity sl1 limit 10`	Configures service policy proximity filtering on the device. • Service policy proximity filtering is only available for wireless clients and is based on Radio Resource 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:**	Specifies an alternate source interface for outgoing mDNS packets on a device.

	Command or Action	Purpose
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. **service-policy-query** {*service-list-name query-period* | **disable**}
7. **redistribute mdns-sd** [**withdraw-only**]
8. **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.

	Command or Action	Purpose
Step 3	**interface** *type number* **Example:**	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	**service-policy-query** {*service-list-name query-period* | **disable**}
	Example:
	`Device(config-if-mdns-sd)# service-policy-query AQ-int 1000`
		Configures periodic browsing of services on an interface or stops browsing of services on an interface.
		Note The difference between the **no** form of this command and the **disable** option is given below:
		• **no** form - If you have enabled browsing of printer services for a specific interface which has a printer connected, and if the printer is removed from the interface, then you can use the **no** form to stop browsing printer services on the interface.
		• **disable** option - Typically, if you have enabled browsing for printer services on the device (globally configured), then printer services are periodically searched for on all the interfaces of the device. If there is an interface where there is no printer service available, you can use the **disable** option to disable browsing of printer services only for the interface.

Step 7	**redistribute mdns-sd** [**withdraw-only**] **Example:** `Device(config-if-mdns-sd)# redistribute mdns-sd withdraw-only`	Speeds up visibility of newly announced services and withdrawal of services when a service or device is turned off. **Note** When you use the **withdraw-only** option, redistribution is only enabled for service withdrawal and not for the service. For example, if service withdrawal announcement for a printer service is enabled, there will not be any announcement about the printer service on other subnets. However, if the printer service is removed, withdrawal announcements will be sent to other devices.

	Command or Action	Purpose
Step 8	**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`	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.

	Command or Action	Purpose
	`service printer-3 regtype _ipp._tcp.local domain tcp4`	
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.
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 Gigabitethernet 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 withdraw-only
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: Extending Services Across Interfaces—Filtering Services per Interface or VLAN

You can filter services available in an interface or VLAN and extend the services (or prohibit the services) to clients in another interface or VLAN, using a single filter command (**match learnt-interface**).

Consider this scenario—In a university setting, the following VLANs are created:

- VLAN 100 contains Apple TVs meant for teachers. These Apple TVs must be inaccessible to students.
- VLAN 200 contains teachers' devices.
- VLAN 300 contains students' devices.
- VLAN 400 contains Apple TVs for teachers and students. These Apple TVs must be accessible to teachers and students.

Extending Apple TV service for teachers—If teachers want to access Apple TVs, then they should be able to send queries for the Apple TV service and receive the Apple TV service from the mDNS cache. A service-list **permit-all** is created to permit all services and queries into, and out of, the device. The service-list configuration is given below:

```
!
service-list mdns-sd permit-all permit 10
!
```

Since there are no restrictions for teachers accessing services on any other VLAN or interface, we can permit all services out of the mDNS cache.

```
!
Interface vlan 200
 description *** Teachers' devices –– Allow all queries (into the cache) and services (from
 the cache).

 service-routing mdns-sd
   service-policy permit-all in
   service-policy permit-all out
   exit
!
```

In the above examples, these actions were taken:

1. The service-list **permit-all** was created to permit queries and services.
2. **permit-all** was applied on VLAN 200 **—**The **permit-all in** command allows all queries into the mDNS cache. Typically, **match** commands are specified for filtering. Since no **match** command is specified, all services or queries are permitted *into* the mDNS cache. Similarly, for the **permit-all out** statement, all services (or queries) are permitted out of the mDNS cache, in response to requests for a service.

Result—Teachers' queries for Apple TV services are allowed out of the subnet and in response, Apple TV service is allowed into the subnet.

Extending Apple TV service for students—Students must be able to send queries for Apple TV and receive responses for accessing Apple TV service. A service-list is created for filtering incoming information (queries coming *into* the cache), and another is created for outgoing information (Apple TV service *from* the cache). The configuration example for creating the service-lists is given below:

```
service-list mdns-sd deny-all permit 10
  match message-type query
service-list mdns-sd deny-all deny 20
```

The service-list **deny-all** permits queries from students (since queries are specified within the 'permit' instruction) and prohibits all other services (since no **match** command is specified within the 'deny' instruction, other types of service information are disallowed.)

```
!
Interface vlan 300
description *** Students' devices — Allow queries from students and nothing else (into the
 cache).

 service-routing mdns-sd
   service-policy deny-all in
   exit
!
```

When the mDNS process encounters the **deny-all in** command, the commands in the service-list **deny-all** are processed sequentially, as given below:

1. 10—Queries for Apple TV service are permitted from the students now.
2. 20—The **deny** command ensures that, apart from queries, no other information is allowed outside the subnet.

The **restricted** service-list is created to permit Apple TV services from VLAN 400 and deny Apple TV services from VLAN 100, meant for teachers.

```
!
service-list mdns-sd restricted permit 10
  match service-type apple-tv
  match learnt-interface vlan400
!
service-list mdns-sd restricted deny 20
  match service-type apple-tv
  match learnt-interface vlan100
!
service-list mdns-sd restricted deny 30
!
```

1. 10—Apple TV services are permitted and all services available in VLAN 400 are permitted.
2. 20—Services available in VLAN 100 are prohibited.
3. 30—All other services are disallowed.

Now, the service filters in the **restricted** service-list are applied onto the students' VLAN, as given below:

```
!
Interface vlan 300
description *** Students' devices — Allow Apple TV services available in VLAN 400. Prohibit
 access of Apple TVs from VLAN 100, and prohibit all other services.

 service-routing mdns-sd
   service-policy restricted out
   exit
!
```

When the mDNS process encounters the **restricted out** command, the commands in the service-list **restricted** are implemented sequentially—Queries for Apple TV services are permitted, Apple TV services from VLAN 400 are permitted for students' access and Apple TV services available in VLAN 100 are prohibited for students' access.

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

Standard/RFC	Title
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 1: Feature Information for Service Discovery Gateway

Feature Name	Releases	Feature Information
Service Discovery Gateway	15.2(1)E	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	Cisco IOS 15.2(2)E	The Service Discovery Gateway feature was enhanced with additional filter and configuration options. In Cisco IOS Release 15.2(2)E, this feature is supported on Cisco Catalyst 3750-X Series Switches. 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**

Feature Name	Releases	Feature Information
Service Discovery Gateway—Phase 3	15.2(3)E	The Service Discovery Gateway feature was enhanced with the following features: • **De-congestion of incoming mDNS traffic using the rate limiting mechanism—**The rate-limit value range was reset to1-100 p/s. • **Redistribution of service-withdrawal announcements across subnets when services are withdrawn, to improve mDNS cache efficiency and to avoid message loops—**The **withdraw-only** option was added to the **redistribute mdns-sd** command. • **A filter criterion for services available and learnt on a specific interface—**The **match learnt-interface** command was added to filter services. • **Enabling and disabling of periodic browsing of services on specific interfaces—**The **service-policy-query (interface)** command was added. For existing, globally configured active queries, the **disable** option was added to disable browsing of services on an interface, retaining the configurations on other interfaces. In Cisco IOS Release 15.2(3)E, this feature is supported on Cisco Catalyst 3750-X Series Switches. The following commands were introduced or modified: **match learnt-interface, rate-limit, redistribute mdns-sd, service-policy-query (interface)**

CHAPTER 2

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

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.

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*] [*mx ns srv*]

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*] [*mx ns srv*]
	Example:
	`Device(config)# ip host vrf vpn3 company1.com 172.16.2.1`
	`Device(config)# ip host test mx 1 mx_record`
	`Device(config)# ip host test ns ns_record`
	`Device(config)# ip host test srv 0 0 0 srv_record`
		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.
		• Mail exchanger (mx) identifies the mail server that is responsible for handling e-mails for a given domain name.
		• Name server (ns) state the authoritative name servers for the given domain.
		• Service (srv) records specifies the location of a service.

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**	Enables privileged EXEC mode.

	Command or Action	Purpose
	Example: `Device> enable`	• Enter your password if prompted.
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 2: Feature Information for VRF-Aware DNS

Feature Name	Releases	Feature Information
VRF-Aware DNS	15.2(1)E	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.