

## Contents

The *Network Registrar CLI Reference Guide* is written for network and system administrators and is intended to provide information about how to use Cisco Network Registrar's command line program, nrcmd.

Use this online guide after you have installed Network Registrar and have it running. This guide provides the following information:

- [Network Registrar CLI Introduction](#)

Provides instructions about how to use the nrcmd program, including batch and interactive operations, command syntax, command attribute guidelines, and navigation.

- [Network Registrar CLI Commands](#)

Gives a detailed description of all the nrcmd commands and their attributes, including usage guidelines and logging information.

- [Using the nrcmd Program in Scripts](#)

Provides suggestions about how to create batch files to execute nrcmd commands.

- [CLI Codes and Formats](#)

Describes status and error codes as well as dump and load formats.

This online guide uses the following notational conventions in command syntax:

Square brackets([ ])—Group optional elements.

Vertical bars (|)—Separate alternative mutually exclusive elements.

Angle brackets (< >)—Indicate you must provide a value for an attribute or argument in the command.

## Network Registrar CLI Introduction

[About the nrcmd Program](#)

[Invoking the nrcmd Command](#)

[Batch Mode](#)

[Interactive Mode](#)

[Registry and Environment Variables](#)

[Command Organization](#)

[Command Usage](#)

[Saving Your Changes](#)

[Refreshing and Clearing the CLI Cache](#)

[Navigation Keys](#)

### About the nrcmd Program

The **nrcmd** command line interface (CLI) enables you to configure and manage your DNS, DHCP, and TFTP servers. This section describes how to use the **nrcmd** CLI. It specifically describes:

- Invoking the command in batch and interactive modes
- Command organization and syntax
- Special keyboard navigation characters

### Invoking the nrcmd Command

You can invoke the **nrcmd** command in batch mode and run scripts that use the commands; or you can invoke the **nrcmd** command in interactive mode and enter commands at the **nrcmd** command prompt. By default, the **nrcmd** command is located in C:\Program Files\Network Registrar\Local\bin on Windows and in /opt/nwreg2/local/usrbin on Solaris and Linux.

---

**Note:** On Windows, if you want to run the **nrcmd** program from outside the installed path, you

must set the CNR\_HOME environment variable.

---

On Windows, you can invoke the **nrcmd** command window from the Start menu:

**Start > Programs > Network Registrar 7.0 > Network Registrar 7.0 CLI**

This method prompts for your user name and password. On Solaris and Linux (as well as Windows alternatively), invoke the command from the command prompt using this syntax:

```
nrcmd [general-options] [command] [options]
```

[Table 1-1](#) describes the general options when invoking from the command prompt.

**Table 1-1 General Options to nrcmd Command**

Option	Description
<b>-C</b> <i>cluster</i>	Cluster ( <i>cluster</i> is the name of the machine on which the Network Registrar servers are running). If not specified, the cluster name defaults to localhost.
<b>-N</b> <i>user</i>	Network Registrar user name ( <i>user</i> ).
<b>-P</b> <i>password</i>	Network Registrar user password ( <i>password</i> ).
<b>-h</b>	Prints help text.
<b>-L</b>	Accesses the local cluster CLI.
<b>-R</b>	Accesses the regional cluster CLI.
<b>-b</b> < <i>file.txt</i>	Batch file ( <i>file.txt</i> is the file of <b>nrcmd</b> commands that run in batch mode, read a line at a time and with a new line printed after the prompt).

---

**Note:** The cluster to which you connect determines the CLI attributes that appear and are available for the release of Network Registrar running on the cluster. This CLI Reference describes the attributes for the current release. For the attributes available for an earlier release, see the CLI Reference for that release.

---

### Batch Mode

The program goes into batch mode if you include a functional command or the **-b** < *file.txt* option on the line. The text file can include any number of **nrcmd** commands, and you can include comment lines preceded by the pound sign (#). In batch mode, you return to the normal system prompt. Note that display in batch mode is intended for parsing by an external program and, therefore, includes only command attributes that have values.

---

**Note:** The last line of code in the input file must end with an end-of-line character. It is also a good practice to make the last line of code an explicit **exit** command.

---

### Interactive Mode

The program goes into interactive mode if you enter just the **nrcmd** command, or include the *cluster*, *user*, or *password* options. To execute the CLI in interactive mode, enter:

```
nrcmd[-C cluster] [-N user] [-P password]
```

This syntax displays the interactive `nrcmd>` prompt, at which you enter a functional command and any optional parameters:

```
nrcmd> command [parameter,parameter, ...]
```

*system-response*

To enter a series of attribute values, insert commas (,) between them. Do not add a space after the comma. If the value is a string containing one or more space characters, enclose the value in quotes:

```
nrcmd> zone example.com. set auth-servers=192.168.50.1,10.0.0.1  
100 Ok  
auth-servers=192.168.50.1,10.0.0.1
```

To terminate an interactive session, enter the **exit** command. To view the online help, enter the **help** command.

## Registry and Environment Variables

If you omit the general options, Network Registrar gets them from the Registry or environment variables. If Network Registrar cannot find values for these parameters, it prompts you for them. If you omit the cluster name on a system where Network Registrar servers are installed, the **nrcmd** program assumes access to **localhost** and does not prompt you.

The environment variables that you can set that are recognized by the **nrcmd** program are CNR\_NAME for the name, CNR\_PASSWORD for the password, and CNR\_CLUSTER for the cluster name.

## Command Organization

The **nrcmd** commands specify a class of objects, which you can create, delete, or list. Each of these objects in turn has attributes, which you can enable, disable, set, get, and unset, depending on data type. These objects may also have common methods, which are specific to the type of object, and that let you perform operations on groups of attributes.

When you use the **nrcmd** commands to configure Network Registrar, you manipulate classes and command attributes.

### Classes

When you use the **nrcmd** commands to configure Network Registrar, you manipulate classes of objects, such as scopes, zones, and servers.

- **create--**Creates an entry. If the entry already exists, this command returns an error.
- **delete--**Removes an entry.
- **list--**Displays all the objects of a given type, including all attributes.

- **listnames**--Displays only the names of all objects of a given type.
- **show**--Displays the values of all the attributes.

### Attributes

- **enable**--Enables a Boolean type of attribute.
- **disable**--Disables a Boolean type of attribute.
- **set**--Sets the value of an attribute.
- **get**--Displays the value of an explicitly defined attribute.
- **unset**--Makes an attribute have no value. You cannot unset required attributes.

---

**Note** You cannot use **nrcmd** to **get** the value of implicitly defined attributes, including implicitly defined default values.

---

There are three ways to set attributes:

- **create** command. For example, to create a High-Availability (HA) DNS server pair, you can specify cluster and IP addresses for the main and backup servers during creation:

```
nrcmd> ha-dns-pair ha-pair-11 create 192.168.50.1 192.168.60.1 main=localhost
backup=backup
```

- Use the **set** or **enable** command after creating the object. For example, you can set just the cluster references to the main and backup server for the created HA DNS pair:

```
nrcmd> < b> ha-pair-1 set main=localhostt backup=backup
```

- Add **attribute=value** pairs at the end of the **create** command.

If you use both the positional value and the **attribute=value** pair for the same attribute on the **create** command line, the **attribute=value** pair is the actual value used (because it comes last).

In interactive mode, all the attributes appear. In batch mode, only those attributes having values appear, and no default values appear. The display in batch mode is less user-friendly, but is more easily parsable by a program. These examples show how output compares in interactive and batch modes, respectively:

```
nrcmd> zone example.com show
```

```
100 Ok
```

```
example.com. (primary):
```

```
checkpoint-interval =
```

```
checkpoint-min-interval =
```

```
defttl = 12h
```

```
dynamic = [default=true]
```

```
dynupdate-set =
```

```
expire = 7d
```

```
...
```

```
$ nrcmd -N admin -P changeme zone example.com show
```

```
100 Ok
```

```
example.com.: defttl=12h; expire=7d; minttl=10m; nameservers={{0 rr2.example.com.}}; ns=rr2.;  
origin=example.com.; person=rr1.; refresh=3h; retry=60m; serial=1; update-acl="key myKey";
```

```
100 Ok
```

- Other custom methods--These are specific operations that you can perform on an object, beyond editing its attributes. Examples are adding a range of IP addresses to a scope, or removing hosts from a zone.

## Command Usage

How you specify a series of arguments depends on the type of command you are using. The following subsections describe the differences between using the **create**, **set**, and **enable** commands.

### Create Keyword

When you use the **create** keyword and there are required arguments, you must supply them. You can also supply additional arguments. You must supply the required arguments in the specified order; however, you can specify the optional arguments in any order with the syntax `attribute=value`.

For example, the syntax for creating a scope is:

```
scopename create ipaddress mask [attribute=value
```

This means that you must supply an IP address and mask when you create a scope, and you can optionally specify other attributes of the scope.

```
scope testScope create 192.168.50.0 255.255.255.0
```

```
100 Ok
```

```
testScope:
```

```
addr = 192.168.50.0
```

```
bootp = disabled
```

```
deactivated =
```

You can also include attribute definitions on the same line. This example creates the same scope, but also specifies the name of the DNS zone to which a DHCP client's host name should be added:

```
scope testscope create
nrcmd> scope testScope create 192.168.50.0 255.255.255.0
100 Ok
testScope:
    addr = 192.168.50.0
    bootp = disabled
    deactivated =
```

After the **create** keyword creates and assigns all specified parameters to the object, it checks that all required attributes have values (either default or user-specified). If you omit required attributes, Network Registrar returns an error.

### Set Keyword

You use the set keyword to set the value of an attribute that is already created. If you want to set a list of values, such as DNS servers or IP addresses, you can separate them with commas.

This example specifies the name of the DNS zone to which a DHCP client's host name should be added:

```
nrcmd> scope testScope set dns-zone-name=example.com.
100 Ok
dns-zone-name=example.com.
```

This example specifies the list of IP addresses for zone transfers for a zone:

```
nrcmd> zone example.com. set auth-servers=192.168.50.1,10.0.0.1
100 Ok
auth-servers=192.168.50.1,10.0.0.1
```

This example sets a client's client-class and domain name:

```
nrcmd client 00:d0:b a:d3:b d:3cbl assesta mc à ì eim t t e r n a l
    domain name = example.com.
100 Ok
```

```
cli nctla sname = internal
domainame = example.com .
```

The **unset** keyword places an attribute in the undefined state. The **get** keyword displays the value for an attribute.

## Enable Keyword

You use the **enable** keyword to enable a Boolean attribute. After you enable one Boolean attribute, you may need to set its associated attributes. Use the **disable** keyword to disable a Boolean attribute. You can use the **unset** keyword to remove the enabled or disabled state of the Boolean attribute.

This example enables incremental transfer processing for the DNS server:

```
checkpoint-min-interval="
nrcmd> checkpoint-min-interval="dns enable ixfr-enable
100 Ok
ixfr-enable=enabled
```

Once incremental transfer is enabled, this example changes its expiration interval:

```
nrcmd> dns set ixfr-expire-interval=10d
100 Ok
ixfr-expire-interval=1w3d
```

---

**Note:** You cannot use **set** and **enable** on the same command line.

---

## Attribute Flags

Command are described as:

- **Required**--The attribute is required for the object, and usually syntactically positional on the **create** command line. You must set the attribute or accept its default, and you can modify the value. You cannot use the **unset** keyword to set a required attribute to undefined. Trying to do so returns the error message 386 - Required attribute cannot be deleted.
- **Optional**--The attribute is optional and does not require a value. You can set and reset the attribute, and you can use the **unset** keyword to make it undefined.
- **Read-only**--The attribute is immutable and read-only. You can use the **get** keyword with the attribute, but you cannot set or unset it. Trying to set or unset a read-only attribute returns the error message 385 - Read-only attribute cannot be modified.

## Saving Your Changes

With new commands introduced in Network Registrar 6.2, **nrcmd** applies the changes you make immediately. (Commands introduced in Network Registrar 6.2 are listed in Release Notes). With the commands from prior releases, the CLI waits for one of these events to occur before it saves your changes to the database:

- Invoking the **save** command
- Exiting from **nrcmd**
- Reloading a server
- Adding a resource record or host to a zone

### Refreshing and Clearing the CLI Cache

The CLI caches many configuration objects that it reads. If multiple users are making changes simultaneously, one CLI instance might have cached an out of date version of an object. The **session cache refresh** command causes the CLI to clear its local cache of all unmodified objects, forcing it to reread objects from the configuration database. The **session cache clear** command forces the CLI to clear all cached data, whether or not unsaved changes were made.

### Navigation Keys

[Table 1-2](#) lists keyboard navigation key combinations that are useful when entering **nrcmd** commands.

**Table 1-2 nrcmd Navigation Key Combinations**

Key Combination	Action
Control-a	Go to the beginning of the line
Control-b	Back one character (or the left arrow key)
Control-d	Delete one character
Control-e	Go to the end of the line
Control-f	Forward one character (or the right arrow key)
Control-k	Kill to the end of the line
Control-l	Redraw the line
Control-n	Next line in the history (or the down arrow key)
Control-p	Previous line in the history (or the up arrow key)
Control-t	Shift an individual character left
Control-u	Delete the line and move the cursor to the beginning of the line
Control-w	Delete one word backwards
Esc-b	Back one word
Esc-f	Forward one word

All contents are Copyright © 1992--2007 Cisco Systems, Inc. All rights reserved.

## Table of Contents

### Overview

- [nrcmd](#)
- [intro](#)
- [expert](#)

### CLI Commands

- [acl](#)
- [addr-trap](#)
- [address-block](#)
- [admin](#)
- [ccm](#)
- [client](#)
- [client-class](#)
- [client-class-policy](#)
- [client-policy](#)
- [cluster](#)
- [dhcp](#)
- [dhcp-address-block](#)
- [dhcp-address-block-policy](#)
- [dhcp-dns-update](#)
- [dhcp-interface](#)
- [dhcp-subnet](#)
- [dns](#)
- [dns-interface](#)
- [dns-update-map](#)
- [exit](#)
- [export](#)
- [extension](#)
- [failover-pair](#)
- [group](#)
- [ha-dns-pair](#)
- [help](#)
- [import](#)
- [key](#)
- [ldap](#)
- [lease](#)
- [lease-notification](#)
- [lease6](#)

- [license](#)
- [link](#)
- [link-policy](#)
- [link-template](#)
- [option](#)
- [option-set](#)
- [owner](#)
- [prefix](#)
- [prefix-policy](#)
- [prefix-template](#)
- [policy](#)
- [region](#)
- [remote-dns](#)
- [report](#)
- [reservation](#)
- [role](#)
- [router](#)
- [router-interface](#)
- [router-type](#)
- [save](#)
- [scope](#)
- [scope-policy](#)
- [scope-template](#)
- [scope-template-policy](#)
- [server](#)
- [session](#)
- [snmp](#)
- [snmp-interface](#)
- [subnet](#)
- [sync-from-dns](#)
- [tftp](#)
- [tftp-interface](#)
- [trap-recipient](#)
- [update-policy](#)
- [vpn](#)
- [zone](#)
- [zone-dist](#)
- [zone-template](#)

## Reference

### nrcmd

nrcmd - run the Network Registrar command line interface

#### Synopsis

```
nrcmd [flags] [<command>]
```

#### Description

Flags are:

```
-C <cluster>    Specify the cluster to connect to.
-N <name>       Specify the administrator name.
-P <password>   Specify the administrator password.
-V <visibility> Specify the visibility level (default 5)
-b             Run in batch mode
-R            Connect to a regional cluster
```

The visibility level controls which properties and features are visible to the session. A lower visibility level makes more properties visible. The default visibility level is 5, and should not be changed without guidance from Cisco technical support.

In batch mode, input is performed a line at a time rather than a character at a time, and the prompt is terminated with a newline character rather than a space.

## Examples

```
% nrcmd -N admin -P changeme -C <local>
nrcmd>
```

## intro

intro - Introduction to nrcmd commands

## Synopsis

## Description

The nrcmd commands fall into two basic groups: regular and irregular. The regular commands manipulate configuration objects such as DHCP Scopes and DNS Zones in a standard fashion. The irregular commands do everything else that is useful. This man page will describe the general pattern of the regular commands. The behavior of the irregular commands will be described in their individual man pages.

### Regular Command form

Regular commands provide common functions for creating, deleting, viewing and editing objects of a given class.

#### Create

```
<cmd> <name> create [<required args>] [<prop>=<val>]
```

#### Delete

```
<cmd> <name> delete
```

#### List

```
<cmd> list
```

#### Modify

```
<cmd> <name> set <prop>=<value> [<prop>=<value> ...]
```

The set command takes two forms: 'set <prop> <value>' for setting a single properties's value, and 'set <prop>=<value> ...' for setting multiple values in a single command.

Errors include:

unknown property  
- if <prop> is not an property name for the object  
invalid format  
- if <value> is not in a valid format  
invalid value  
- if <value> is not semantically valid

<cmd> <name> get <prop>  
The get command returns the value of the named property.

<cmd> <name> unset <prop> [<prop> ...]  
The unset command makes the named properties have no value.

<cmd> <name> enable <feature>  
The enable command sets the value of the named feature to true.

<cmd> <name> disable <feature>  
The enable command sets the value of the named feature to false.

<cmd> <name> show  
The show command displays the value of the object.

#### Class specific commands (methods)

The configuration behavior of some objects may be enhanced by the addition of class specific commands to perform a useful action such as modifying complex properties, or controlling the objects behavior.

For example, DHCP Scope objects contain lists of address ranges from which leases may be offered. To manipulate this list of ranges, the scope command provides the commands: addRange, removeRange, and listRanges.

Another example is the forceAvailable command provided by the lease command to tell the DHCP server that a given lease should be forced into the available state.

#### Filters

The results of the list commands can be restricted by applying filters to the results. The filter specification is similar to the LDAP query filter specification, but uses infix rather than prefix relational operators. See the filter man page for a more complete description of the filter specification grammar.

#### Format specifiers

The output format of the list and show commands can be specified with the format arguments. See the format man page for a complete description of the format specifiers.

#### Licensing

nrcmd requires the current cluster to have a valid license. If the license is invalid or has expired, only the 'license' command will be operational; it may be used to establish a new license key. A warning will be issued upon login if the license will be expiring within 7 days.

#### Locking

nrcmd versions prior to CNR 6.2 are limited to a single session and use a locking mechanism to control access.

In interactive mode, the program attempts at startup to get an exclusive lock for the cluster to which it connects. If this fails, nrcmd will issue a warning, and will allow only the following commands to be executed: "client", "lease", "zone create", "help", and "force-lock". The force-lock command obtains the exclusive lock indiscriminately, and should be used with caution. In particular, it should not be used unless you're sure that no one else is updating the cluster.

The warning issued for interactive mode, when the lock cannot be obtained, is

```
408 Already locked: '<user>@<host>.<pid>'.
Warning: unable to lock the cluster.
You might want to use the force-lock command if
you're sure that no one else is updating the cluster
```

where

```
<user> is the name of the administrator
<host> is the name of the Unix or Windows host on which the
        other user is running
<pid>  is the process ID of the nrcmd or
        Network Registrar process. (You can use system
        tools such as Unix "ps" or NT "Task Manager" to
        determine if the process is still running.)
```

In batch mode (when the command is entered on the command line), nrcmd will not attempt to obtain a lock unless the command is not one of the valid ones listed above. If a lock is needed, it will then attempt to get it; if this fails, the error issued is

```
408 Cannot lock cluster: Already locked: '<user>@<host>.<pid>
```

where <user>, <host>, and <pid> are as above.

#### Return codes

All nrcmd commands will return a status code as the first line of output. The status codes are heavily influenced by SMTP and other line oriented protocols. The first word of the line is a three digit status code, and the remaining words on the line are descriptive text that may or may not be constant for a given status code. The first digit of the status code determines the class of the status:

```
1xx - the command completed successfully, possibly with warnings
3xx  there was some error in processing the command
4xx  errors in communicating with the cluster database server
5xx  there is was an internal error in the program
```

#### Property types

The properties that are manipulated by the set and get command have specific data types which determine the syntactically valid values. These types are:

```
AT_STRING - a string, valid inputs are:
            * any text

AT_INT    - an integer, valid inputs are:
            * decimal digits, or
            * 0x followed by hex digits.
```

AT\_BOOL - a boolean value, valid inputs are:  
 \* true, on, enabled, 1, or  
 \* false, off, disabled, 0.

AT\_DATE - a date, valid inputs are:  
 \* 'forever'  
 \* +<time value>

AT\_TIME - a span of time, in seconds, valid inputs are:  
 \* decimal number of seconds  
 \* combination of numbers of weeks, days, hours, minutes, and seconds for example: 1w2d3h4m5s.

AT\_IPADDR - an ip address, valid inputs are:  
 \* dotted quad format, for example 10.24.1.2

AT\_MACADDR - a MAC address, valid inputs are:  
 \* raw hex digits, for example: 010203040506  
 \* hex digits separated by ':', '.', or '-', for example: 01:02:03:04:05:06, 01-02-03-04-05-06, 01.02.03.04.05.06  
 \* type and length, followed by hex digits, for example: 1,6,ab:01:cd:02:ef:03

AT\_RANGEINT - a range restricted integer  
 AT\_RANGETIME - a range restricted time value  
 AT\_ENUMINT - an enumerated integer  
 AT\_FLAGSINT - a bitmask with named bit positions

#### Validation

Data validation will be done at configuration creation and property modification time. The nrcmd CLI will check for required valid values when a configuration object is created, and it will check the validity of property values when they are set.

Dangling references that are created by deleting a referred-to object, such as the policy for a scope, or the client class for a client will not be caught by the CLI.

## Examples

## Limitations

## expert

Expert mode commands

## Synopsis

**cluster** <local-cluster> **delete**

**ccm sync-from-dhcp** [**Reservation**|**AddressSpace**|**ALL**]

**ccm sync-to-dhcp** [**Prefix**|**Link**|**FailoverPair**|**Reservation**|**ALL**]

**ccm sync-from-dns** [**ZoneData**|**Hosts**]

**dhcp setFailoverState** <state> <partner-name> <main | backup>

```
dhcp disableComm <partner-name>
dhcp enableComm

reservation sync [-to|-from] dhcp

sync-from-dns
```

## Description

These commands are available only in expert mode and must be used with care. To enter expert mode:

```
nrcmd> session set visibility=3
```

The local cluster can only be deleted while in expert mode.

The ccm sync-from-dhcp command can be used to synchronize CCM data from the DHCP server configuration data.

The ccm sync-to-dhcp command can be used to synchronize the DHCP server's data from CCM's data.

The ccm sync-from-dns command can be used to synchronize CCM DNS zone and RR data or hosts from RR data from DNS. The sync-from-dns, which is retained for backwards compatibility, is the same as ccm sync-from-dns ZoneData.

The dhcp commands can be used to force failover related actions. These must be used with extreme care and are not recommended.

The object command can be used to display the object with a specified oid. If -class=<classname> is specified, the DB for that class is used (the classname must be specified using the correct case). If -db=<dbid> is specified, the specified DB is used (the dbid must be specified in uppercase). If neither -class or -db is specified, the CCM DB is assumed.

The reservation sync command can be used to synchronize reservations between CCM and DHCP. These are identical to the ccm sync-from-dhcp/sync-to-dhcp reservation commands.

## Examples

## Limitations

## acl

acl - Manages DNS access control lists which are used to control zone access for DNS updates, zone transfers and queries

## Synopsis

```
acl list
acl listnames
acl <name> show
acl <name> create "<match-list>"
acl <name> delete
acl <name> get <attribute>
```

```
acl <name> set <attribute>=<value>
acl <name> unset <attribute>
acl <name> add [!][key] <value>
acl <name> remove [!][key] <value>
```

## Description

The `acl` command is used to manage DNS ACLs which are used to restrict dynamic DNS updates, zone transfers and queries. Once you have created the `acl` object, it can be used with the `update-acl`, `restrict-xfer-acl` and `restrict-query-acl` on the DNS server or a zone object.

You can specify the `match-list` as a comma-separated list of values, enclosed in quotes, or you can use the `add` and `remove` commands to edit the match list. The `add` command will add elements to the end of the list. The `remove` command will remove the first matching element in the `match-list`.

Match list entries can consist of IP node or subnet addresses, TSIG keys, or ACLs. A TSIG key must also be preceded by the keyword `<key>`. The `<!>` notation can be used to negate an entry in the list.

## Examples

```
nrcmd> acl my-acl create "key my-key, 10.1.0.0/16"
nrcmd> acl my-acl set match-list="10.1.1.1/32, my-acl"
nrcmd> acl my-acl add "!10.2.0.0/16"
```

## Status

## See Also

[key](#)

## Attributes

### `match-list` [amelist](#)

Displays a comma-separated list of match elements, which can consist of IP node or subnet addresses, TSIG keys, or ACLs. You can also use the following reserved words as elements in a match list:

```
any
none
localhost
localnets
```

To specify more than one element in the match list, enclose the list in quotation marks; for example:

```
"192.168.2.1, localhost"
```

Use the object name to reference another ACL or TSIG key. A key must also be preceded by the keyword `<>key<>`;

```
for example,
key mykey.
```

You must specify subnet addresses in address/mask format. Use an exclamation point (!) to negate an entry in the list; for example,

```
"!192.168.3.0/24, !youracl"
```

Note: You can define the name reference to an ACL in the match list before you actually create the ACL. But the ACL must exist before you start or reload the DNS server. If the DNS cannot resolve an ACL name reference on either the DNS server object or a zone object, it will flag the error and will not

## addr-trap

addr-trap - Configures free-address monitoring by the DHCP server

### Synopsis

```
addr-trap <name> create
addr-trap <name> delete
addr-trap enable <attribute>
addr-trap <name> disable <attribute>
addr-trap <name> set <attribute>=<value> [<attribute>=<value> ...]
addr-trap <name> unset <attribute>
addr-trap <name> get <attribute>
addr-trap list | listnames
addr-trap <name> [show]
```

### Description

The addr-trap command configures values that the DHCP server uses to monitor free-address levels. Use this command with the SNMP server to provide SNMP notification trap messages as free-address levels change within the DHCP server.

### Examples

### Status

### See Also

[scope](#), [trap-recipient](#)

### Attributes

**enable** [bool](#) default = on

Indicates whether this configuration is active, causing scopes to track their free-address levels and possibly send trap even

**high-threshold** [percent](#) default = 25%

Sets the level at which the 'low' threshold will be re-enabled and the high-threshold trap will be generated.

See the low-threshold for details on how the free-address level is calculated.

**low-threshold** [percent](#) default = 20%

Sets the free-address level at which a low-threshold trap will be generated, and the 'high' threshold will be re-enabled.

For scopes, the free-address level is calculated as follows

$$100 * \frac{\text{available non-reserved leases}}{\text{total configured leases}}$$

where the counts are the sum across all of the scopes included in the aggregation.

For prefixes, the free-address level is calculated as follows

```

max-leases - dynamic leases
100 * -----
max-leases

```

where the counts are the sum across all of the prefixes included in the aggregation.

**mode** `enumint`(scope=1, network=2, selection-tags=3, prefix=4, link=5, v6-selection-tags=6, countonly=7, v6-countonly=8) default = scope

Indicates how scopes should aggregate their free address level. The 'scope' mode causes each scope to track its own free-address level independently. This is an IPv4 only mode.

The 'network' mode causes all of the scopes configured by the object to aggregate their free-address levels if they share 'primary-subnet'. This is an IPv4 only mode.

The 'selection-tags' grouping causes scopes to aggregate their free-address information together if they share a primary-subnet and if their lists of selection tags match exactly. This is an IPv4 only mode.

The 'prefix' mode causes each prefix to track its own free-address level independently. This is an IPv6 only mode.

The 'link' mode causes all of the prefixes configured by the object to aggregate their free-address levels if they share a link. This is an IPv6 only mode.

The 'v6-selection-tags' grouping causes prefixes to aggregate their free-address information if they share a link, and if their lists of selection tags match exactly. This is an IPv6 only mode.

The 'countonly' and 'v6-countonly' groupings are used for the built-in aggregation objects that are used to provide top-utilized information when no traps are configured.

No traps will be fired for these modes.

**name** `string` required,unique,immutable

Gives a unique name to the configuration object.

## address-block

`address-block` - Defines a contiguous block of IP address space

### Synopsis

```

address-block <address/mask-length> create[<attribute>=<value> ...]
address-block <address/mask-length> delete
address-block list
address-block listnames
address-block <name> show
address-block <name> get <attribute>
address-block <name> set <attribute>=<value> [<attribute>=<value> ...]
address-block <name> unset <attribute>

```

### Description

An address block is an aggregate of IP addresses based on a power-of-two address space. For example, the 192.168.0.0/16 address block includes 65536 ( $2^{16}$ ) addresses.

Address blocks can be further divided into child address blocks and subnets. For example, you might want to divide the 192.168.0.0/16 address block further into four child address blocks: 192.168.0.0/17, 192.168.64.0/17, 192.168.128.0/17, and 192.168.192.0/17. A subnet is used to designate a leaf node of the address space that will not be further subdivided.

Address blocks are used as a management tool to group and report on address space usage. The owner and region properties on an address block or its parent can be used to constrain user access to address space reports.

## Examples

```
nrcmd> address-block 192.168.0.0/16 create
```

## Status

## See Also

[subnet](#), [owner](#), [region](#)

## Attributes

### **address** [subnet](#) required,immutable

Specifies the IP address and mask of the CCM address block which was set at creation. This defines the address range of the block. Use the set command to redefine the address.

### **description** [string](#)

Describes how this address block is used.

### **forward-zone-name** [dname](#)

Names the forward zone associated with this block.

### **owner** [oid](#)

Names the owner of this address block. Use the owner field to group similarly owned address blocks; to limit administrative access; and to track allocation or delegation for ARIN reporting purposes.

### **parent** [oid](#)

Identifies the parent address block.

### **region** [oid](#)

Names the region associated with this address block. Use the region field to group similarly located address blocks to limit administrative access.

### **report-state** [enumint](#)(available=0, internal=1, reallocated=2, reassigned=3)

Transient report state of the CCM address block, provided to make it easier for ARIN reporting and for clients to filter the list of address blocks they display.

```
0 available
1 internal
2 reallocated
3 reassigned
```

### **reverse-zone-name** [dname](#)

Names the reverse zone associated with this block.

### **sink** [oid](#)

Points to either the CCMOwner or CCMCluster for a leaf block delegated to a lower-level sink. Delegated AddrBlocks (those with a non-null sink attribute) cannot be further split into child address blocks or sub-

### **source** [oid](#)

Points to the CCMOwner configuration object representing

the source for a top-level block allocated from a high source.

**type** [nameref](#)(CCMAddrSpaceType)

Defines an addresss-block type, enabling you to group address blocks that share common properties, such as the associated scope template. This attribute names a CCMAddrSpaceType object that contains more information the type.

**vpn-id** [int](#) default = 0, immutable

Identifies the ID the VPN object used to support multiple address spaces, such as in a managed VPN environment.

## admin

admin - Creates administrators and assigns them groups and passwords

### Synopsis

```
admin <name> create [<attribute>=<value>]
admin <name> delete
admin list
admin listnames
admin <name> set <attribute>=<value> [<attribute>=<value> ...]
admin <name> get <attribute>
admin <name> unset <attribute>
admin <name> enable <attribute>
admin <name> disable <attribute>
admin <name> show
admin <name> enterPassword
```

### Description

The admin command configures administrators for the cluster.

You can choose any string for the name of the administrator. Names are not case-sensitive. Network Registrar uses a password to authenticate each administrator. Passwords are case-sensitive.

Because the password is sensitive information, Network Registrar prints its value as '\*\*\*\*\*'.

```
admin <name> enterPassword
```

If you want to enter a password and not have Network Registrar display the password on your screen, create an administrator but do not supply a password. Then use the enterPassword command to have Network Registrar prompt you twice for the password. If both entries match, Network Registrar will set the password value.

### Examples

### Status

### See Also

[group](#), [role](#)

### Attributes

**groups** [string](#)

Lists the names of groups to which this administrator belongs.

**password** [clrtxt](#) **transient**

Temporarily stores the cleartext password for this administrator that is used to create the password-secret.

**superuser** [bool](#)

Indicates whether this administrator is a superuser.

## ccm

ccm - Configures and controls the CCM server

### Synopsis

```
ccm get <attribute>
ccm set <attribute>=<value> [<attribute>=<value> ...]
ccm unset <attribute>
ccm show
```

### Description

The ccm command manages the CCM server in the cluster.

### Examples

### Status

### See Also

[server](#)

### Attributes

**atul-port** [rangeint](#)(1-65535) default = 7543

Enables the CCM server to listen for incoming ATUL queries on this UDP port. This attribute is used only by the regional CCM server. Changes to this setting take effect on the next server restart.

**atul-support** [bool](#) default = disabled

Enables CCM support for the ATUL query protocol. The atul-port parameter configures the port number to listen on for ATUL queries. Changes to this setting take effect on the next server restart.

**lease-hist-detail** [bool](#) default = true

If polling for lease history data, this causes CCM to ask for history detail data when polling DHCP servers, and save the detail data if it's returned. Changes to this setting will take effect on the next server restart.

**local-zone-edit-mode** [enumint](#)(staged=1, synchronous=2) default = synchronous

Indicates the default mode that local clients should use for DNS updates. This default mode is overridden if a mode is specified in a given SCP message (in other words, by specific client request). If unset, clients should always present the choice of mode to the user.

**poll-lease-hist-interval** [rangetime](#)(0-1y) default = 4h

This interval specifies how often to collect the lease history information from all the DHCP servers. If set to 0, polling is disabled.

**poll-lease-hist-offset** [rangetime\(0-24h\)](#)

Provides a fixed time of day for lease history polling. This time is interpreted as a time of day offset, with 0 being 12 midnight, provided the polling interval is less than 24 hours and the offset value is less than the polling interval. If the c value is greater than the polling interval, or the interval is greater than 24 hours, the offset will be ignored. The scheduler for polling will ensure that the first polling event occurs at the offset time. For example, if you set the interval 4 hours and the offset to 2am, the polling would occur at 2am, 6 10am, 2pm, 6pm and 10pm.

**poll-lease-hist-retry** [rangeint\(0-4\)](#) default = 1

The number of retries for a given polling interval, if polling f

**poll-replica-interval** [rangetime\(0-1y\)](#) default = 4h

This interval specifies the default value of how often to poll for configuration changes when replicating data from a local clu

**poll-replica-offset** [rangetime\(0-24h\)](#) default = 0

Provides a fixed time of day for replica polling. This time is interpreted as a time of day offset, with 0 being 12 midnight, provided the polling interval is less than 24 hours and the offset value is less than the polling interval. If the c value is greater than the polling interval, or the interval is greater than 24 hours, the offset will be ignored. The scheduler for polling will ensure that the first polling event occurs at the offset time. For example, if you set the interval 4 hours and the offset to 2am, the polling would occur at 2am, 6 10am, 2pm, 6pm and 10pm.

**poll-subnet-util-interval** [rangetime\(0-1y\)](#) default = 4h

This interval specifies how often to collect subnet utilization from all the DHCP servers. If set to 0, polling is disabled.

**poll-subnet-util-offset** [rangetime\(0-24h\)](#)

Provides a fixed time of day for subnet utilization polling. This time is interpreted as a time of day offset, with 0 being 12 midnight, provided the polling interval is less than 24 hours and the offset value is less than the polling interval. If the c value is greater than the polling interval, or the interval is greater than 24 hours, the offset will be ignored. The scheduler for polling will ensure that the first polling event occurs at the offset time. For example, if you set the interval 4 hours and the offset to 2am, the polling would occur at 2am, 6 10am, 2pm, 6pm and 10pm.

**poll-subnet-util-retry** [rangeint\(0-4\)](#) default = 1

The number of retries for a given polling interval, if polling f

**poller-event-threads** [rangeint\(1-5\)](#) default = 2

This specifies how many threads that the poller will create. Changes to this setting will take effect on the next server rest

**regional-zone-edit-mode** [enumint\(staged=1, synchronous=2\)](#) default = staged

Indicates the default mode that regional clients should use for DNS updates. This default mode is overridden if a mode is specified in a give SCP message (in other words, by specific client request). If

unset, clients should always present the choice of mode to the u

**scope-edit-mode** `enumint(staged=1, synchronous=2)` default = staged

Indicates the default mode that web UI and CLI clients use for DHCP edits:

- 1 staged  
Edits are written to the database but are not immediately forwarded to the DHCP server; that is, they remain unpublished by DHCP.
- 2 synchronous  
Edits are written to the database and are immediately forwarded for publishing to the DHCP server.

**trim-changeset-age** `rangetime(24h-3y)` default = 1y

Sets the minimum age for a change log entry to be eligible for trimming.

**trim-lease-hist-age** `rangetime(24h-1y)` default = 24w

The minimum length of time that we will keep a lease history record in the lease history database. Any lease history record older than this time will be deleted when the next lease history database trimming operation occurs.

If set to 0, all lease history data will be trimmed. Changes to this setting will take effect on the next server restart.

**trim-lease-hist-interval** `rangetime(0-1y)` default = 24h

This interval specifies how often to trim the old lease history records. If set to 0 no automatic lease history trimming occurs.

If lease history collection and polling are enabled and this parameter is set to 0, the lease history database will continue to grow without bound. Changes to this setting will take effect on the next server restart.

**trim-subnet-util-age** `rangetime(24h-1y)` default = 24w

The age that we will allow a subnet-util trimming element to be eligible for deleting. If this value is set to zero, we will not consider the record age when trimming. Changes to this setting will take effect on the next server restart.

**trim-subnet-util-interval** `rangetime(0-1y)` default = 24h

This interval specifies how often to trim the old subnet utilization data. If set to 0 no automatic subnet utilization trimming occurs. Changes to this setting will take effect on the next server restart.

**webui-mode** `enumint(Basic=0, Advanced=1, Expert=2)` default = Basic

Sets the mode for the web UI at session startup:

- 0 basic
- 1 advanced
- 2 expert

## client

`client` - Creates clients and assigns them to client-classes

### Synopsis

```
client <name> create [<attribute>=<value>]  
client <name> delete  
client list  
client listnames  
client <name> show  
client <name> get <attribute>  
client <name> set <attribute>=<value>  
client <name> unset <attribute>
```

## Description

The `client` command assigns attributes to a specific client entry. These attributes determine what type of IP address, policy, or both that Network Registrar assigns to the requesting host. Network Registrar always stores the client identifier (MAC address or default) in lowercase characters.

Because the DHCP server reads the client-specific configuration information each time a request comes in, you do not have to reload the server after modifying it. However, you must reload the server if you modify the default client configuration.

The attributes you can assign include such things as a class of client, a policy, an action, and the inclusion or exclusion of scope selection tags. The DHCP server looks up these properties to determine how it should process a host request for an IP address.

If you have common client attributes to configure, such as selection criteria, use the `client-class` so that multiple client configurations can reference the attributes.

You can specify the client by using the MAC address or some other unique client identifier related to the `client-lookup-id` that is specified in the client's associated `client-class`.

A sample Ethernet MAC address might be `1,6,00:a0:24:2e:9c:20`

```
client name create [attribute=value...]
```

```
client default create [attribute=value...]
```

Creates the client identifier as a MAC address or the word `default` (and optionally defines its attributes). The default client configuration applies to all clients that do not have an explicit configuration. If an entry for the client already exists, the command overwrites it.

If using a MAC address, it should be in the form `hardware, length, address` (without spaces and including the commas):

`hardware`

Usually 1 (Ethernet) or 6 (Token Ring), but can be any number from 1 thr

`length`

Octets in the MAC address (usually 6, but can be any number from 1 throu

`address`

MAC address itself, with octets separated by colons, and each octet havi two-character hex value from 00 through FF (not case-sensitive).

## Examples

```
nrcmd> client 1,6,00:d0:ba:d3:bd:3b create client-class-name=external
```

## Status

## See Also

[client-class](#)

## Attributes

**action flags**(exclude=1, deprecated-one-shot=2, deprecated-use-release-grace-period=3, none=32)

Describes the action the DHCP server takes for this client-class. The flags are:

- 1 exclude - causes the server to ignore all communication from this client.
- 2 deprecated-one-shot - now deprecated.
- 3 deprecated-use-release-grace-period - now deprecated.
- 32 none

If you specify the exclude action in the default client-class entry, then any client not specifically registered through the client command cannot communicate with the server. Note: The deprecated flags (2,3) are now available through the policy command attributes inhibit-all-renews and release-grace-period.

**add-to-environment-dictionary** *string*

Lists attribute-value pairs that are added to the environment dictionary whenever this client-class is associated with an incoming DHCP request. You can use these attribute-value pairs to configure extensions or expressions without having the executable code in the extension or expression.

The string must have the format:

```
"attribute1=value1,attribute2=value2, ... ,attributen=valuen"
```

**authenticate-until** *date*

Sets an authentication expiration date, using date forever keyword. Dates can be in the 2h (two hours ago) example) or month day hour:minute[:second] year format. Formats for the date are:

+

Time in the future, where num is a decimal number. num is s, m, h, d, or w for seconds, minutes, hours, days, weeks, respectively.

[::]

Month, day, 24-hour time, and 2-or-4-digit-year.

For example: Jun 30 20:00:00 2007. Enter the time in local to the nrcmd process.

forever

Does not expire the authentication for this client-class.

**client-class-name** *string*

Identifies the client-class to which a client belongs. If a client is not a member of a client-class, then the DHCP server uses the default client-class properties.

**default-vpn** *nameref*(VPN)

Names the VPN to assign to clients that do not already have a vpn-id or vrf-name value.

**domain-name** *string*

Gives the domain name (which must be a zone) to use with DNS updates. Places the client's A record in this DNS zone. This feature is maintained for compatibility with previous versions. Additional options to specify the forward zone are provided through the client policy (or embedded policy) and its referer.

DNSUpdateConfig objects.

**embedded-policy** [obj\(0\)](#)

Specifies the embedded policy object for this client.

**host-name** [string](#)

Specifies the hostname. Use this string to replace any DHCP option that the DHCP client sends. The two forms specifying the hostname are:

1. A string that does not start with an at (@) sign. The value of host-name is used to override the DHCP client request host name. When you enter a valid name, you tell the DHCP server to ignore any host-name specified for this client, and, instead, use this client-entry attribute value. The actual value of the hostname option in the client packet is ignored. You can use any valid DNS name, but cannot use underscores (\_) in the hostname.
2. A string that starts with an at (@) sign. Network uses this form of hostname value to signal the following special handling:

`@host-name-option`

Causes the server to use whatever hostname option the client sent. This is the default behavior if there is no entry for hostname in either the client or client-class object.

`@no-host-name-option`

Causes the server to drop the hostname option from the client sent, and not replace it. If you have dynamic DNS name synthesis, then the client will have no name in DNS.

`@use-macaddress`

Causes the server to synthesize a hostname for this client that is derived from its MAC address, and thus unique. This token is used to ensure that this client has a valid name in DNS.

This feature is maintained for compatibility with earlier versions. Additional options for hostname synthesis are provided in the DNSUpdateConfig object referenced by the policy hierarchy.

**over-limit-client-class-name** [string](#)

Identifies which client-class to use if this client is over the limit allowed for the number of simultaneous active leases. This is a common limitation-id.

**override-vpn** [nameref\(VPN\)](#)

Names the VPN to assign to clients, no matter what value is present as VPN-IDs or vrf-names.

**policy-name** [string](#)

Identifies the policy to add to Network Registrar's DNS search list for this client.

**selection-criteria** [string](#)

Lists selection tags for this client. All the criteria in the list must appear in the scope/prefix selection tags for this client to be considered acceptable to this client.

**unauthenticated-client-class-name** [string](#)

Identifies the client-class to use if this client is not authenticated.

**user-defined** [string](#)

Contains an opaque user-defined string that can be set and queried. This attribute has no effect on the operation.

DHCP server.

## client-class

client-class - Creates client-classes

### Synopsis

```
client-class <name> create [<attribute>=<value>]
client-class <name> delete
client-class list
client-class listnames
client-class <name> show
client-class <name> set <attribute>=<value> [<attribute>=<value> ...]
client-class <name> get <attribute>
client-class <name> unset <attribute>
```

### Description

The client-class command applies a set of attributes to a group or class of DHCP client configurations. Unlike most client configurations, the DHCP server reads the client-class configurations at server startup time. Therefore, you must reload the server for changes to take effect.

You must enable client-class processing for the server for Network Registrar to recognize client-classes, as the examples show.

### Examples

```
nrcmd> dhcp enable client-class
nrcmd> client-class internal create
nrcmd> dhcp reload
```

### Status

### See Also

### Attributes

**action flags**(exclude=1, deprecated-one-shot=2, deprecated-use-release-grace-period=3, deprecated-inhibit-all-renews=4, deprecated-release-grace-period=5, none=32)

Specifies what action to take with this client-class. You can specify exclude, causing the server to ignore all communication from this client.

If you specify the exclude action in the default client-class, then any client not specifically registered through the client-class command cannot communicate with the server.

The one-shot and use-release-grace-period actions are now deprecated and ignored; these are available through the deprecated-inhibit-all-renews and deprecated-release-grace-period attributes.

**add-to-environment-dictionary string**

This string contains attribute-value pairs that are added to the environment dictionary whenever this client-class is associated with an incoming DHCP request. You can use these attribute-value pairs to configure extensions or expressions without having to re-write the executable code in the extension or expression. The string must have the format:

"attribute1=value1,attribute2=value2, ... ,attributen=va

**client-lookup-id** *expr*

Specifies the key value used to lookup the specified client database, using an expression that evaluates to a string or a blob that is a valid string. The lookup can be local or LDAP.

**default-vpn** *nameref(VPN)*

Determines the VPN to which a client is assigned if the client does not supply a VPN-ID (or vrf-name) value.

**domain-name** *string*

Sets the domain name, which must be a zone, for performing DNS updates. Places the client's A record in this DNS domain. This feature is maintained for compatibility with prior versions. Additional options to specify the forward zone are provided. The client-class's policy (or embedded policy) and its referenced DNSUpdateConfig objects.

**embedded-policy** *obj(Policy)*

Specifies the embedded policy object for this client-class.

**host-name** *string*

Specifies the hostname for clients in this client-class. Network Registrar uses this form of hostname value to provide the following special handling:

*@host-name-option*

Causes the server to use whatever hostname option the client sent. This is the default behavior if there is no entry for host-name in either the client or client-class object.

*@no-host-name-option*

Causes the server to drop the hostname option that the client sent, and not replace it. If you have disabled DNS name synthesis, then the client will have no hostname placed into DNS.

*@use-macaddress*

Causes the server to synthesize a host-name for the client that is derived from its MAC address, and is unique. This token is used to ensure that a client has a valid name in DNS.

This feature is maintained for compatibility with prior versions. Additional options for host-name synthesis are provided in the DNSUpdateConfig object referenced by the policy hierarchy.

**limitation-id** *expr*

Specifies an expression that evaluates to a binary large object (blob), or a string that can be used as a blob. The result value groups leases that have a maximum limit on the number of simultaneous active leases allowed. To configure the limitation-count attribute of the policy command. See also the over-limit-client-class attribute.

**over-limit-client-class-name** *string*

Designates the client-class used if this client is over the limit allowed for the specified limitation-id.

**override-client-id** *expr*

Specifies the client-identity value for the specified client-class. It is an expression that evaluates to a binary large object (blob) or a value that is derived from the expression evaluation result. The client-id option value in the incoming packet (though in practice, both values are retained in the lease-state data

**override-vpn [nameref](#)(VPN)**

Determines the VPN to which a client is assigned regardless of what the client provides for a VPN-ID (or vrf-name) value.

**policy-name [string](#)**

Names the policy that should be used for this client-class.

**selection-criteria [string](#)**

Lists the selection tags for this client-class. All the tags in this list must appear in the scope/prefix selection tags for a scope/prefix to be considered acceptable to this client-class.

**unauthenticated-client-class-name [string](#)**

Names the client-class used if this client is no longer authenticated.

**user-defined [string](#)**

Provides an opaque user-defined string that can be set and queried. This property has no effect on the operation of the DHCP server.

**v6-client-lookup-id [expr](#)**

Specifies the key value used to lookup the DHCPv6 client database, using an expression that evaluates to a string or a blob that is a valid string. The lookup can be local or LDAP.

**v6-override-client-id [expr](#)**

An expression which evaluates to a blob which is used for the client-identity value for the current DHCPv6 client. Both this value derived from the expression evaluation will replace the client-id option value in the incoming packet (though in practice, both values are retained in the lease-state database).

## client-class-policy

`client-class-policy` - Adds DHCP policy information to a client-class

### Synopsis

```
client-class-policy <client-class-name> delete
```

```
client-class-policy <client-class-name>
```

```
    set <attribute>=<value> [<attribute>=<value> ...]
```

```
client-class-policy <client-class-name> get <attribute>
```

```
client-class-policy <client-class-name> disable <attribute>
```

```
client-class-policy <client-class-name> enable <attribute>
```

```
client-class-policy <client-class-name> show
```

```
client-class-policy <client-class-name> setLeaseTime <time-val>
```

```
client-class-policy <client-class-name> getLeaseTime
```

```
client-class-policy <client-class-name> setOption <opt-name | id> <value>
```

```
client-class-policy <client-class-name> getOption <opt-name | id>
```

```
client-class-policy <client-class-name> unsetOption <opt-name | id>
```

```
client-class-policy <client-class-name> listOptions
```

```
client-class-policy <client-class-name> setV6Option <opt-name | id> <value>
```

```
client-class-policy <client-class-name> getV6Option <opt-name | id>
```

```
client-class-policy <client-class-name> unsetV6Option <opt-name | id>
```

```
client-class-policy <client-class-name> listV6Options
```

```

client-class-policy <client-class-name>
    setVendorOption <opt-name | id> <opt-set-name> <value>

client-class-policy <client-class-name>
    getVendorOption <opt-name | id> <opt-set-name>

client-class-policy <client-class-name>
    unsetVendorOption <opt-name | id> <opt-set-name>

client-class-policy <client-class-name> listVendorOptions

client-class-policy <client-class-name>
    setV6VendorOption <opt-name | id> <opt-set-name> <value>

client-class-policy <client-class-name>
    getV6VendorOption <opt-name | id> <opt-set-name>

client-class-policy <client-class-name>
    unsetV6VendorOption <opt-name | id> <opt-set-name>

client-class-policy <client-class-name> listV6VendorOptions

```

## Description

The `client-class-policy` command configures embedded policies for client-classes. Each client-class can contain option data in its embedded policy and can refer to a named policy with more option data; for example, a router IP address.

An embedded policy is a collection of DHCP option values and settings associated with (and named by) a client-class. Network Registrar implicitly creates and deletes an embedded client-class policy when you create and delete the corresponding client-class. You manipulate the client-class policy using the name of the client-class to which the embedded policy is attached.

```

client-class-policy <client-class-name> setOption <opt-name | id> <value>
    Sets individual option values. When you set an
    option value the DHCP server will replace any existing value or
    create a new one as needed for the given option name.

```

```

client-class-policy <client-class-name> getOption <opt-name | id>
    Displays option values.

```

```

client-class-policy <client-class-name> unsetOption <opt-name | id>
    Unsets option values.

```

```

client-class-policy <client-class-name> setLeaseTime <time-val>
    Sets the lease time.

```

```

client-class-policy <client-class-name> getLeaseTime
    Displays the lease time for the specified client-class.

```

See the attribute descriptions for the policy command for a complete list of attributes for `client-class-policy`. Except where noted in that list, policy command attributes also apply to client-class policies.

## Examples

## Status

## See Also

[policy](#), [scope-policy](#), [client-policy](#)

## Attributes

### **affinity-period** *time*

Associates a lease in the AVAILABLE state with the client that last held the lease. If the client requests a lease during the affinity period, it is granted the same lease; that is, renewals are prohibited, then it is explicitly not given. Because of the vast IPv6 address space and depending on the generation technique, it could be millions of years before the same address ever needs reassignment to a different client, so there is no reason to hold on to this information for that long. To prohibit renewals enable either the inhibit-all-renewals or the inhibit-renews-at-reboot attribute.

### **allow-client-a-record-update** *bool* default = disabled

Determines if a client is allowed to update A records. If the client sets the flags in the FQDN option to indicate it wants to do the A record update in the request, and the allow-client-a-record-update value is TRUE, the server allows the client to do the A record update; otherwise, based on other server configurations the server does the A record update.

### **allow-dual-zone-dns-update** *bool* default = disabled

Enables DHCP clients to perform DNS updates into two DNS zones. To support these clients, you can configure the DHCP server to allow the client to perform an update, but also to perform the update on the client's behalf.

### **allow-lease-time-override** *bool* default = disabled

Gives the server control over the lease period. Although a client can request a specific lease time, the server need not honor the request if this attribute is set to false (the default). Even if set to true, clients can request only lease times that are shorter than those configured for the server.

### **allow-non-temporary-addresses** *bool* default = true

Determines whether DHCPv6 clients can request non-temporary (IA\_NA) addresses.

The default is to allow clients to request non-temporary addresses.

### **allow-rapid-commit** *bool* default = false

Determines whether DHCPv6 clients can use a Solicit with the Rapid Commit option to obtain configuration information with fewer messages. To permit this, make sure that a single server is servicing clients.

This attribute has special handling during the policy hierarchy processing when checking the Prefix policies (embedded in the Prefixes on a Link. The Prefixes for the Link are processed in alphabetic (case blind) order. For each Prefix, the embedded and then named policy are checked. Only Prefix policies the client has access (based on selection tags, etc.) are checked.

- If any of the prefix policies has this attribute set to FALSE, Rapid Commit is not allowed.
- If at least one has it set to TRUE, Rapid Commit is allowed.
- Otherwise, the remaining policies in the hierarchy are checked.

checked.

The default is not to allow clients to use Rapid Commit

**allow-temporary-addresses** `bool` default = true

Determines whether DHCPv6 clients can request temporary addresses.

The default is to allow clients to request temporary ad

**default-prefix-length** `rangeint(0-128)` default = 64

For delegation, specifies the default length of the del if a router (client) does not explicitly request it.

The default length must always be less than or equal to length of the prefix range.

**forward-dnsupdate** `nameref(DnsUpdateConfig)`

Specifies the name of the update configuration that det which forward zones to include in updates.

**forward-zone-name** `dname`

Designates an optional forward zone for DNS updates.

**giaddr-as-server-id** `bool` default = false

Enables the DHCP server to set the server-id option on a and a DHCPACK to the giaddr of the incoming packet, inst IP address of the server (the default action).

This causes all unicast renews to be sent to the relay a of directly to the DHCP server, and so renews arrive at server with option-82 information appended to the packe Some relay agents may not support this capability and, : complex configurations, the giaddr might not actually be to which the DHCP client can send A unicast packet. In t the DHCP client cannot renew a lease, and must always p rebind operation (where the DHCP client broadcasts a re of sending a unicast to what it believes is the DHCP se:

**grace-period** `time` default = 5m

Defines the length of time between the expiration of a : and the time it is made available for reassignment.

**inhibit-all-renews** `bool` default = false

Causes the server to reject all renewal requests, forcin to obtain a different address any time it contacts the l

**inhibit-renews-at-reboot** `bool` default = false

Permits clients to renew their leases, but the server fo them to obtain new addresses each time they reboot.

**limitation-count** `int`

Specifies the maximum number of clients with the same l: that are allowed to have currently active and valid lea:

**offer-timeout** `time` default = 2m

Instructs the server to wait a specified amount of time has offered a lease to a client, but the offer is not y At the end of the specified time interval, the server ma lease available again.

**packet-file-name** `string`

Identifies the boot-file to use in the boot process of : The server returns this file name in the 'file' field o: The packet-file-name cannot be longer than 128 characte:

**packet-server-name** `string`

Identifies the host-name of the server to use in a clien process. The server returns this file name in the 'sname

of its replies. The packet-server-name field cannot be more than 64 characters.

### **packet-siaddr** **ipaddr**

Identifies the IP address of the next server in the client process. For example, this might be the address of a TFTP server used by BOOTP clients. The server returns this address in the 'siaddr' field of its replies.

### **permanent-leases** **bool** default = disabled

Indicates whether leases using this policy are permanent to requesting clients. If leases are permanently granted, the server ignores the configured dhcp-lease-time option value and maximum lease time.

### **preferred-lifetime** **time** default = 1w

Assigns the default and maximum preferred lifetime for DHCPv6 client interfaces. Expressed in seconds and relative to the time the server sent the packet, this attribute sets the length of time that the address is preferred; that is, the preferred lifetime is unrestricted. When the preferred lifetime expires, the address becomes deprecated and its use is restricted.

### **reconfigure** **enumint**(allow=1, disallow=2, require=3) default = allow

Controls DHCPv6 client reconfiguration support:

- |            |                                                                                                                                                               |
|------------|---------------------------------------------------------------------------------------------------------------------------------------------------------------|
| 1 allow    | Allows clients to request reconfiguration support and the server will honor the request (default).                                                            |
| 2 disallow | Allows clients to request reconfiguration support but the server will not honor the clients' request.                                                         |
| 3 require  | Requires clients to request reconfiguration support and the server drops client Solicit and Request messages that do not include a Reconfigure-Accept option. |

This attribute has special handling during the policy hierarchy processing when checking the Prefix policies (embedded in the Prefixes on a Link. The Prefixes for the Link are processed in alphabetic (case blind) order. For each Prefix, the embedded and then named policy are checked. Only Prefixes for which the client has access (based on selection tags, etc.) are checked as follows:

- If any of the prefix policies has this attribute set to disallow or require, that setting is used.
- Otherwise, if at least one has it set to allow, Reconfigure is allowed.
- If no prefix policies have this attribute set, the policies in the hierarchy are checked.

### **reconfigure-via-relay** **bool** default = false

Controls whether the server should prefer unicasting or relaying DHCPv6 Reconfigure messages.

If false (the default), the server prefers to unicast Reconfigure messages if the client has one or more valid statefully assigned addresses.

If true, the server prefers to send Reconfigure messages via the relay agent unless no relay agent information is available.

Note: When you use this attribute, consider that:

- In networks where the DHCPv6 server cannot communicate directly with its client devices?for example, where firewalls are in place?set this value to true.

- The DHCPv6 server does not use embedded and named policies configured on a client when it evaluates this attribute.
- The relay agent cannot be used if the Relay-Forward came from a link-local address.

### **reverse-dnsupdate** [nameref\(DnsUpdateConfig\)](#)

Specifies the name of the update configuration that determines which reverse zones to include in a DNS update.

### **server-lease-time** [time](#)

Tells the server how long a lease is valid. For more information on communication with a client, you might have the server offer longer leases as leased for a longer period than the client configured. This also provides more lease-time stability. This value is used unless it is longer than the lease time in the dhcp-lease-time found through the normal traversal of policies.

### **split-lease-times** [bool](#) default = disabled

Specifies a value that the DHCP server might use internally to affect lease times.

If enabled, the DHCP server still offers clients lease times that reflect the configured lease-time option from the appropriate policy; but the server bases its decisions regarding expiration on the 'server-lease-time' value.

### **unavailable-timeout** [time](#) default = 24h

Permits the server to make a lease unavailable for the specified time and then to return the lease to available state. If the attribute is configured in the system\_default\_policy, then the default is 86400 seconds (or 24 hours).

### **use-client-id-for-reservations** [bool](#) default = off

Controls how the server database checks for reserved IP addresses.

By default, the server uses the MAC address of the DHCP client key for its database lookup. If this attribute is set to on (enabled), then the server does the check for reserved IP addresses using the DHCP client-id, which the client supplies. In cases where the DHCP client does not supply the client-id, the server synthesizes it, and uses that value.

### **v4-bootp-reply-options** [optionid4](#)

Lists the options the server returns to all BOOTP clients.

### **v4-reply-options** [optionid4](#)

Lists the options the server returns to all DHCPv4 clients, or not the client specifically asks for the option data.

### **v6-reply-options** [optionid6](#)

Lists the options that should be returned in any replies to DHCPv6 clients.

This attribute has special handling during the policy handling process when checking the Prefix policies (embedded in the Prefixes on a Link. The Prefixes for the Link are processed in alphabetic (case blind) order. For each Prefix, the embedded and then named policy are checked. Only Prefixes for which the client has access (based on selection tags, etc.) are returned.

### **valid-lifetime** [time](#) default = 2w

Assigns the default and maximum valid lifetime for leases on DHCPv6 client interfaces. Expressed in seconds and relative to the time the server sent the packet, this attribute specifies the length of time that an address remains valid. When

period of time expires, the address becomes invalid and unusable. The valid lifetime must be greater than or equal to the preferred lifetime.

## client-policy

`client-policy` - Adds DHCP policy information to a client object

### Synopsis

```
client-policy <client-name> delete
client-policy <client-name> set <attribute>=<value> [<attribute>=<value> ..
client-policy <client-name> get <attribute>

client-policy <client-name> disable <attribute>
client-policy <client-name> enable <attribute>
client-policy <client-name> show

client-policy <client-name> setLeaseTime <time-val>
client-policy <client-name> getLeaseTime

client-policy <client-name> setOption <opt-name | id> <value>
client-policy <client-name> getOption <opt-name | id>
client-policy <client-name> unsetOption <opt-name | id>
client-policy <client-name> listOptions

client-policy <client-name> setV6Option <opt-name | id> <value>
client-policy <client-name> getV6Option <opt-name | id>
client-policy <client-name> unsetV6Option <opt-name | id>
client-policy <client-name> listV6Options

client-policy <client-name>
  setVendorOption <opt-name | id> <opt-set-name> <value>

client-policy <client-name>
  getVendorOption <opt-name | id> <opt-set-name>

client-policy <client-name>
  unsetVendorOption <opt-name | id> <opt-set-name>

client-policy <client-name> listVendorOptions

client-policy <client-name>
  setV6VendorOption <opt-name | id> <opt-set-name> <value>

client-policy <client-name>
  getV6VendorOption <opt-name | id> <opt-set-name>

client-policy <client-name>
  unsetV6VendorOption <opt-name | id> <opt-set-name>

client-policy <client-name> listV6VendorOptions
```

### Description

The `client-policy` command configures embedded policies for clients. Each client can contain option data in its embedded policy and might refer to a named policy with more option data—for example, a router IP address. Network Registrar implicitly creates and deletes an embedded client policy when you create or delete the corresponding client. You manipulate the client policy using the name of the client

to which the embedded policy is attached.

**client-policy** <client-name> **setOption** <opt-name | id> <value>  
Sets individual option values. When you set an option value the DHCP server replaces any existing value or creates a new one as needed for the given option name.

**client-policy** <client-name> **unsetOption** <opt-name | id>  
Unsets option values.

**client-policy** <client-name> **getOption** <opt-name | id>  
Displays option values for the specified client.

**client-policy** <client-name> **setLeaseTime**  
Sets lease time values for the specified client.

**client-policy** <client-name> **getLeaseTime**  
Displays the lease time value.

## Examples

## Status

## See Also

[policy](#), [scope-policy](#), [client-class-policy](#)

## Attributes

### **affinity-period** *time*

Associates a lease in the AVAILABLE state with the client that last held the lease. If the client requests a lease during the affinity period, it is granted the same lease; that is, renewals are prohibited, then it is explicitly not given. Because of the vast IPv6 address space and depending on the address generation technique, it could be millions of years before an address ever needs reassignment to a different client, so there is no reason to hold on to this information for that long. To prohibit renewals enable either the `inhibit-all-renewals` or the `inhibit-renews-at-reboot` attribute.

### **allow-client-a-record-update** *bool* default = disabled

Determines if a client is allowed to update A records. If the client sets the flags in the FQDN option to indicate that it wants to do the A record update in the request, and the `allow-client-a-record-update` value is TRUE, the server allows the client to do the A record update; otherwise, based on other server configuration, the server does the A record update.

### **allow-dual-zone-dns-update** *bool* default = disabled

Enables DHCP clients to perform DNS updates into two DNS zones. To support these clients, you can configure the DHCP server to allow the client to perform an update, but also to perform the update on the client's behalf.

### **allow-lease-time-override** *bool* default = disabled

Gives the server control over the lease period. Although a client can request a specific lease time, the server need not

request if this attribute is set to false (the default). Even if set to true, clients can request only lease times shorter than those configured for the server.

**allow-non-temporary-addresses** **bool** default = true

Determines whether DHCPv6 clients can request non-temporary (IA\_NA) addresses.

The default is to allow clients to request non-temporary

**allow-rapid-commit** **bool** default = false

Determines whether DHCPv6 clients can use a Solicit with Rapid Commit option to obtain configuration information with fewer messages. To permit this, make sure that a single server is servicing clients.

This attribute has special handling during the policy list processing when checking the Prefix policies (embedded in the Prefixes on a Link). The Prefixes for the Link are processed in alphabetic (case blind) order. For each Prefix embedded and then named policy are checked. Only Prefixes the client has access (based on selection tags, etc.) are

- If any of the prefix policies has this attribute set to FALSE, Rapid Commit is not allowed.
- If at least one has it set to TRUE, Rapid Commit is allowed.
- Otherwise, the remaining policies in the hierarchy are checked.

The default is not to allow clients to use Rapid Commit

**allow-temporary-addresses** **bool** default = true

Determines whether DHCPv6 clients can request temporary addresses.

The default is to allow clients to request temporary addresses

**default-prefix-length** **rangeint**(0-128) default = 64

For delegation, specifies the default length of the delegation if a router (client) does not explicitly request it.

The default length must always be less than or equal to the length of the prefix range.

**forward-dnsupdate** **nameref**(DnsUpdateConfig)

Specifies the name of the update configuration that designates which forward zones to include in updates.

**forward-zone-name** **dname**

Designates an optional forward zone for DNS updates.

**giaddr-as-server-id** **bool** default = false

Enables the DHCP server to set the server-id option on the relay and a DHCPACK to the giaddr of the incoming packet, instead of the IP address of the server (the default action).

This causes all unicast renews to be sent to the relay agent or directly to the DHCP server, and so renews arrive at the server with option-82 information appended to the packets. Some relay agents may not support this capability and, in complex configurations, the giaddr might not actually point to which the DHCP client can send a unicast packet. In the DHCP client cannot renew a lease, and must always perform a rebind operation (where the DHCP client broadcasts a request of sending a unicast to what it believes is the DHCP server).

**grace-period** **time** default = 5m

Defines the length of time between the expiration of a lease and the time it is made available for reassignment.

**inhibit-all-renews** **bool** default = false

Causes the server to reject all renewal requests, forcing clients to obtain a different address any time it contacts the server.

**inhibit-renews-at-reboot** **bool** default = false

Permits clients to renew their leases, but the server will not allow them to obtain new addresses each time they reboot.

**limitation-count** **int**

Specifies the maximum number of clients with the same IP address that are allowed to have currently active and valid leases.

**offer-timeout** **time** default = 2m

Instructs the server to wait a specified amount of time before it has offered a lease to a client, but the offer is not guaranteed. At the end of the specified time interval, the server will make the lease available again.

**packet-file-name** **string**

Identifies the boot-file to use in the boot process of the client. The server returns this file name in the 'file' field of its replies. The packet-file-name cannot be longer than 128 characters.

**packet-server-name** **string**

Identifies the host-name of the server to use in a client's boot process. The server returns this file name in the 'server-name' field of its replies. The packet-server-name field cannot be longer than 64 characters.

**packet-siaddr** **ipaddr**

Identifies the IP address of the next server in the client's boot process. For example, this might be the address of a TFTP server used by BOOTP clients. The server returns this address in the 'siaddr' field of its replies.

**permanent-leases** **bool** default = disabled

Indicates whether leases using this policy are permanent for requesting clients. If leases are permanently granted, the server ignores the configured dhcp-lease-time option value and uses the maximum lease time.

**preferred-lifetime** **time** default = 1w

Assigns the default and maximum preferred lifetime for DHCPv6 client interfaces. Expressed in seconds and relative to the time the server sent the packet, this attribute specifies the length of time that the address is preferred; that is, the address is unrestricted. When the preferred lifetime expires, the address becomes deprecated and its use is restricted.

**reconfigure** **enumint**(allow=1, disallow=2, require=3) default = allow

Controls DHCPv6 client reconfiguration support:

- |   |          |                                                                                                                                                               |
|---|----------|---------------------------------------------------------------------------------------------------------------------------------------------------------------|
| 1 | allow    | Allows clients to request reconfiguration support and the server will honor their request (default).                                                          |
| 2 | disallow | Allows clients to request reconfiguration support but the server will not honor the clients' request.                                                         |
| 3 | require  | Requires clients to request reconfiguration support and the server drops client Solicit and Request messages that do not include a Reconfigure-Accept option. |

This attribute has special handling during the policy processing when checking the Prefix policies (embedded in the Prefixes on a Link). The Prefixes for the Link are processed in alphabetic (case blind) order. For each Prefix,

embedded and then named policy are checked. Only Prefix policies that the client has access (based on selection tags, etc.) are checked, as follows:

- If any of the prefix policies has this attribute set to disallow or require, that setting is used.
- Otherwise, if at least one has it set to allow, Relay-For-wards is allowed.
- If no prefix policies have this attribute set, the policies in the hierarchy are checked.

### **reconfigure-via-relay** **bool** default = false

Controls whether the server should prefer unicasting or relaying DHCPv6 Reconfigure messages.

If false (the default), the server prefers to unicast Reconfigure messages if the client has one or more valid statefully assigned addresses.

If true, the server prefers to send Reconfigure messages via the relay agent unless no relay agent information is available.

Note: When you use this attribute, consider that:

- In networks where the DHCPv6 server cannot communicate directly with its client devices?for example, with firewalls in place?set this value to true.
- The DHCPv6 server does not use embedded and named policies configured on a client when it evaluates this attribute.
- The relay agent cannot be used if the Relay-For-wards came from a link-local address.

### **reverse-dnsupdate** **nameref**(DnsUpdateConfig)

Specifies the name of the update configuration that determines which reverse zones to include in a DNS update.

### **server-lease-time** **time**

Tells the server how long a lease is valid. For more information on communication with a client, you might have the server offer leases as leased for a longer period than the client could use. This also provides more lease-time stability. This value is used unless it is longer than the lease time in the dhcp-lease-time found through the normal traversal of policies.

### **split-lease-times** **bool** default = disabled

Specifies a value that the DHCP server might use internally to affect lease times.

If enabled, the DHCP server still offers clients lease times that reflect the configured lease-time option from the appropriate policy; but the server bases its decisions regarding lease renewal on the 'server-lease-time' value.

### **unavailable-timeout** **time** default = 24h

Permits the server to make a lease unavailable for the specified time and then to return the lease to available state. If this attribute is configured in the system\_default\_policy, then the default value is 86400 seconds (or 24 hours).

### **use-client-id-for-reservations** **bool** default = off

Controls how the server database checks for reserved IP addresses.

By default, the server uses the MAC address of the DHCP client as the key for its database lookup. If this attribute is set to on (enabled), then the server does the check for reserved IP addresses using the DHCP client-id, which the client uses to identify itself.

In cases where the DHCP client does not supply the client ID, the server synthesizes it, and uses that value.

#### **v4-bootp-reply-options** [optionid4](#)

Lists the options the server returns to all BOOTP clients.

#### **v4-reply-options** [optionid4](#)

Lists the options the server returns to all DHCPv4 clients, or not the client specifically asks for the option data.

#### **v6-reply-options** [optionid6](#)

Lists the options that should be returned in any replies to DHCPv6 clients.

This attribute has special handling during the policy processing when checking the Prefix policies (embedded in the Prefixes on a Link. The Prefixes for the Link are processed in alphabetic (case blind) order. For each Prefix, the embedded and then named policy are checked. Only Prefixes for which the client has access (based on selection tags, etc.) are checked.

#### **valid-lifetime** [time](#) default = 2w

Assigns the default and maximum valid lifetime for leased DHCPv6 client interfaces. Expressed in seconds and relative to the time the server sent the packet, this attribute specifies the length of time that an address remains valid. When the period of time expires, the address becomes invalid and unusable. The valid lifetime must be greater than or equal to the preferred lifetime.

## cluster

`cluster` - Configures the local and remote clusters

### Synopsis

```
cluster <name> create <address> [<attribute>=<value>]  
cluster <name> delete  
cluster list  
cluster listnames  
cluster <name> show  
cluster <name> set <attribute>=<value> [<attribute>=<value> ...]  
cluster <name> get <attribute>  
cluster <name> unset <attribute>  
  
cluster <name> enable <attribute>  
cluster <name> disable <attribute>
```

### Description

The `cluster` command configures the specified local or remote cluster, primarily providing connection and polling information; for example, IP address, fully qualified domain name, and HTTP port. This information provides reference points for objects related to the cluster.

### Examples

### Status

### See Also

## Attributes

**admin** *string*

Sets the administrator identity to use to contact this cluster.

**atul-port** *rangeint(0-65535)*

if > 0, a process on this cluster is listening for ATUL queries on this port number.

**cluster-id** *short unique,immutable*

Identifies the local cluster that is the authoritative source for this object. This attribute is set as part of replica propagation.

**fqdn** *dname*

Provides the fully qualified domain name of this server. This attribute is not used to contact the cluster.

**http-port** *int*

Specifies the http-port to use for non-SSL-secured connections to the web server for this cluster.

**https-port** *int*

Controls the https-port to use for SSL-secured connections the webserver for this cluster. This port is only used if value of the use-https-port attribute is true.

**ipaddr** *ipaddr required,unique*

Provides the IP address of this server. This attribute, rather than the fqdn is used to connect to the cluster.

**local-servers** *obj(CCMServer) transient*

Lists the servers associated with this cluster. This transient attribute makes it easier for clients that want to show a list of clusters with their child servers to get all the information in a single request.

**name** *string required,unique*

Names the cluster.

**password** *clrtxt transient*

Sets the password that authenticates the identity stored in the admin attribute. Do not use this clear-text value except within process memory. Use the corresponding password-secret instead.

**password-secret** *secret*

Sets the identifier of the secret representing the password that authenticates the identity stored in the admin attribute.

**poll-lease-hist-interval** *rangetime(0-1y)*

Specifies how often to collect the lease history from the IIS server for this cluster. If set to 0, polling does not occur.

**poll-lease-hist-offset** *rangetime(0-24h)*

Provides a fixed time of day for lease history polling. This time is interpreted as a time of day offset, with 0 being 12 midnight, provided the polling interval is less than 24 hours and the offset value is less than the polling interval. If the value is greater than the polling interval, or the interval is greater than 24 hours, the offset will be ignored. The scheduler for polling will ensure that the first poll occurs at the offset time. For example, if you set the interval to 12 hours and the offset to 10:00 AM, the first poll will occur at 10:00 AM on the first day.

4 hours and the offset to 2am, the polling would occur at 2 10am, 2pm, 6pm and 10pm.

**poll-lease-hist-retry rangeint(0-4)**

Controls how often to retry if lease history polling fails.

**poll-replica-interval time default = 4h**

Set the automatic replication interval; that is, how often Network Registrar polls this server for replica data.

**poll-replica-offset time default = 4h**

Sets a fixed time of day for replica polling.

This time is interpreted as a time of day offset, with 0 be 12 midnight, provided that:

the polling interval is less than 24 hours, and

the offset value is less than the polling interval.

If the offset value is greater than the polling interval, or the interval is greater than 24 hours, the offset is ignored.

The scheduler for polling ensures that the first polling event occurs at the offset time. For example, if you set the interval to 4 hours and the offset to 2am, the polling would occur at

2am, 6am, 10am, 2pm, 6pm and 10pm.

**poll-subnet-util-interval rangetime(0-1y)**

Specifies how often to collect subnet utilization data from the DHCP server for this cluster. If set to 0, polling does not occur.

**poll-subnet-util-offset rangetime(0-24h)**

Provides a fixed time of day for subnet utilization polling.

This time is interpreted as a time of day offset, with 0 be 12 midnight, provided the polling interval is less than 24

and the offset value is less than the polling interval. If the value is greater than the polling interval, or the interval is

greater than 24 hours, the offset will be ignored.

The scheduler for polling will ensure that the first polling event occurs at the offset time. For example, if you set the interval

4 hours and the offset to 2am, the polling would occur at 2 10am, 2pm, 6pm and 10pm.

**poll-subnet-util-retry rangeint(0-4)**

Controls how often to retry if subnet utilization polling fails.

**product-version pcv**

The product version number of the cluster in major, minor, patch

This value is updated when the cluster is resynchronized.

**remote-id short**

Sets the ID on the remote cluster that refers back to the local cluster. If there are two cluster objects on two servers

that share a secret and refer to each other, then the local ID = the remote ID, and the local remote ID = the remote ID, and the local shared secret = the value the remote shared secret

of the local shared secret = the value the remote shared secret

**replication-initialized bool default = false**

Indicates whether data replication has already been initialized on this cluster.

**restore-state enumshort default = active, transient**

Indicates whether the cluster has been deactivated or is in process of being restored from the replica db.

**scp-port int**

Controls the port number used for SCP communications.

### **scp-read-timeout** **time** default = 20m

The time limit for how long we should wait for data when reading an SCP message from this cluster.

### **shared-secret** **secret**

Specifies the identifier for the secret shared between the server storing this object and the cluster it represents. This shared secret is used to generate single-sign-on authentication tokens.

### **use-https-port** **bool** default = false

Controls whether the https-port is used to make single sign connections to the cluster. If the value is false, or the https-port attribute is not set, then the http-port is used

### **use-ssl** **enumstr**(disabled=1, optional=2, required=3) default = optional

What security mode should we use when connecting to this cluster. If optional, and we have a security library installed, we will try to secure the connection. If required, we will not make the connection unless we can secure the connection (this requires the security library to be installed). If none, we will not try to secure the connection.

## dhcp

dhcp - Configures and controls the DHCP server

### Synopsis

```
dhcp disable <attribute>
dhcp enable <attribute>

dhcp get <attribute>
dhcp set <attribute>=<value> [<attribute>=<value> ...]
dhcp unset <attribute>
dhcp show

dhcp getStats [[all | server [,] failover [,] dhcpv6] [total | sample]]
dhcp resetStats
dhcp getScopeCount [FailoverPair <name> | vpn <name> | all]
dhcp getPrefixCount [vpn <name> | all]

dhcp attachExtension <extension-point> <extension-name> [sequence number]
dhcp detachExtension <extension-point> [sequence number]
dhcp listExtensions

dhcp setPartnerDown <partner-server-name> [<date>]
dhcp getRelatedServers [column-separator=<string>]

dhcp updateSms [all]
dhcp serverLogs show
dhcp serverLogs nlogs=<nlogs> logsize=<logsize>

dhcp limitationList <ipaddr> [<limitation-id>] show
```

### Description

The dhcp command lets you configure the DHCP server in a cluster.

```
dhcp getStats [[all | server [,] failover [,] dhcpv6] [total | sample]]
dhcp resetStats
```

The getStats command retrieves statistics from a running DHCP

server. You can supply one or more specific categories of statistics counters, or the keyword all to retrieve all supported categories. If collection of sample counters is enabled in the server, you can retrieve the most recent sample counters instead of the running totals by specifying sample after the categories. The resetStats command resets the running totals counters.

**dhcp attachExtension** <extension-point> <extension-name> [sequence number]

**dhcp detachExtension** <extension-point> [sequence number]

**dhcp listExtensions**

Use the commands attachExtension, detachExtension, and listExtensions to configure the extensions points in the server.

You can associate multiple extensions with each extension point, and each executes in the order specified by the sequence number used when the attachment was made. If no sequence number is used with attachExtension and detachExtension, it defaults to 1. If multiple extensions are configured for a given point, listExtensions shows the sequence numbers associated with each. Sequence numbers must be in the range 1-32.

The available extension points are:

The attachExtension command sets the specified extension point (and optional sequence position) to call the named extension. If the extension point is already configured (for a given sequence position) to call an extension, Network Registrar overwrites it with the new value.

The detachExtension command removes any extension configuration from the specified extension point and sequence number.

The listExtensions command shows the current configurations for each extension point.

You can put the DHCP server into import mode by enabling the import-mode feature and then restarting the server. You take the server out of import-mode by disabling the feature and restarting the server. You can use import mode to exclude all DHCP lease requests except for the specially tagged ones that come from the CLI during lease import (see the import command).

**dhcp setPartnerDown** <partner-server-name> [<date>]

The dhcp setPartnerDown command notifies the DHCP server that one of its safe failover partner servers is down. The date specified represents a time equal to or later than the last known time the partner server could have been operational. If no date is specified, the current time is used. The time value should be entered using the local time of the nrcmd process. Formats for the date are:

-<num><value>

where <num> is a decimal number and <value> is one of 's', 'm', 'h', 'd', 'w', in which 's' is seconds, 'm' is minutes, 'h' is hours, 'd' is days and 'w' is weeks.

<month> <day> <hour>:<minute>[:<second>] <year>

where <month> is the name or first three letters of the name of the month, <hour> is the hour on a 24- hour clock, and <year> is

the fully-specified year or a two-digit representation in which  
98 = 1998, 99 = 1999 and all other two digit values XX = 20XX.

**dhcp getRelatedServers [column-separator=<string>]**

The dhcp getRelatedServers command displays a table with the following information for each associated safe failover, DNS or LDAP server:

Type  
Main, Backup, DNS or LDAP

Name  
DNS host name

Address  
IP Address in dotted octet format

Communications  
OK or INTERRUPTED

Requests  
Number of outstanding requests

<cluster-name> State  
Failover state of this server

Partner State  
Failover state of partner

**dhcp updateSms [all]**

The dhcp updateSms command initiates SMS processing. To send all leases to SMS, use the argument all; otherwise, only the new leases activated since the last time the command ran successfully are sent. To run this command, turn on sms-network-discovery and set sms-library-path. The command returns an error if sms-network-discovery is not turned on or if it is unable to load SMS library or if the optional argument string is invalid, otherwise it returns success to indicate SMS processing started successfully.

**dhcp serverLogs show**

The serverLogs show command displays the number of log files and the maximum size for each file.

The serverLogs command allows setting the two server logging parameters, nlogs and logsize. Either or both may be specified in the command, and changes will only occur to the one(s) specified. When setting logsize, the value may be suffixed with K or M to signify units of thousands or millions. Note that in order for these changes to take effect you must save the changes and restart the server Agent.

```
dhcp serverLogs nlogs=6 logsize=500K
dhcp serverLogs logsize=5M
```

**dhcp getScopeCount [FailoverPair <name> | vpn <name> | all]**

The getScopeCount command displays the scopes, networks, and VPNs for the current VPN, all VPNs, a specific VPN, or a failover pair. The getPrefixCount command displays the prefixes, links, and VPNs

for the current VPN, all VPNs, or a specific VPN.

If the ip-history feature is enabled, you should trim records from the history database to reclaim disk space. Each history record has a binding end time. Periodically, the DHCP server examines the lease history records, and deletes any records for bindings which ended at least ip-history-max-age in the past.

**dhcp limitationList <ipaddr> [<limitation-id>] show**

Lists DHCP clients and leases that are associated by a common limitation-id for the client (see the client command). Use this command when a DHCP client is denied service because the number of existing clients with a common limitation-id equals the allowed limitation-count, as set for a policy (see the policy command). It then determines which existing clients with that limitation-id have active leases.

If you specify both the ipaddr and limitation-id arguments, the ipaddr determines the network in which to search, and does not have to be an actual IP address that the DHCP server could allocate. In this case, the limitation-id must be a blob in nn:nn:nn format (such as 01:02:03) or a string in string format. If you omit the limitation-id, the ipaddr must be the IP address of a currently active lease, and the limitation-id used for the command will be the one associated with that lease.

If you want to determine the existing clients and leases using up the limitation-count for a particular limitation-id because the following message appeared in the DHCP server log:

```
Warning Server 0 05646 Could not add Client MAC:
'1,6,01:02:03:04:0c:03' with limitation-id: 01:02:03
using Lease: 10.0.0.23, already 3 Clients with that id.
No over-limit client class specified! Dropping packet!
```

Use the lease specified in "... using Lease 10.0.0.23" as the <ipaddr>, and the limitation-id specified in "... with limitation-id 01:02:03" as the <limitation-id>:

```
nrcmd> dhcp limitationList 10.0.0.23 01:02:03 show
```

The result would be a list of 3 leases with the client's MAC address, the client last transaction time, and the client's host name.

## Examples

## Status

## See Also

[server](#)

## Attributes

### **activity-summary-interval** *time* default = 5m

Sets the time that elapses between activity summary log messages. You must also enable the activity-summary sett

in log-settings.

**addr-blocks-default-selection-tags** [string](#)

Associates a default selection tag (or list of tags) with subnet-allocation requests that do not contain any subnet data.

**addr-blocks-use-client-affinity** [bool](#) default = true

Controls whether the DHCP server allocates subnets to clients using address-blocks that the clients have already used. If you set this attribute to false, the server allocates from any suitable block (based on other selection data in messages).

**addr-blocks-use-lan-segments** [bool](#) default = false

Controls whether DHCP subnet-allocation uses the lan-seg attribute when it is configured on address-blocks.

**addr-blocks-use-selection-tags** [bool](#) default = true

Controls whether the server compares subnet name data on incoming subnet-allocation request with the selection tag address-block. A block is only considered if the two match.

**atul-port** [rangeint](#)(1-65535) default = 7543

If configured, enables the DHCP server to listen for incoming ATUL queries on this UDP port.

**atul-support** [bool](#) default = off

Enables DHCP support for the ATUL query protocol. The atul parameter configures the port number to use to listen for ATUL queries.

**client-cache-count** [int](#) default = 1000

Specifies the maximum number of clients in the client cache. The DHCP server allocates the amount at startup and frees it at shutdown. If you set the value to 0, you disable client caching and the server must use persistent storage to process DHCPREQUEST.

**client-cache-ttl** [time](#) default = 10s

Sets the maximum time-to-live in seconds for a client in cache. The DHCP server discards the entries in memory after this period.

**client-class** [bool](#) default = disabled

Controls how the DHCP server uses the client and client-configuration objects to affect request processing. Default is false (disabled).

**client-class-lookup-id** [expr](#)

Specifies the expression used to determine a client-class on data contained in an incoming DHCP client request. The expression must return a string with the name of a configured client-class, otherwise it must return the string ''. Any return that is not a string containing the name of a currently configured client-class or '' is considered an error.

**cnr-5-0-upgraded** [date](#)

Indicates whether the DHCP server was upgraded from 3.5 Read-only.

**collect-addr-util-duration** [rangetime](#)(0-3d) default = 0

Sets the maximum period, in hours, that the DHCP server maintains address utilization data. To disable collecting address utilization data, unset

this attribute or give it a value of 0.  
Use this attribute, with the collect-addr-util-interval attribute, to determine the frequency that the server uses to take snapshots of the data and to set the length of time the DHCP server maintains the data. Together these have an impact on memory usage because each snapshot of 68 bytes.

For example, if you have 10 scopes and you set collect-addr-util-duration is set to 24h and collect-addr-util-interval to 1h, memory used by the DHCP server to maintain address utilization data is 24 times for each scope; that is, 10x24x68 or 16KB.

### **collect-addr-util-interval** [rangetime\(5m-24h\)](#) default = 15m

Sets the frequency, in minutes or hours, that the DHCP server uses to take snapshot views of address utilization. If collect-addr-util-duration is not configured, or set to 0, the DHCP server ignores the collect-addr-util-interval attribute, with the collect-addr-util-duration attribute, to determine the frequency that the server uses to take snapshots of the data and the length of time the DHCP server maintains the data. Together these attributes affect memory usage because each snapshot of data is 68 bytes.

For example, if you have 10 scopes and you set collect-addr-util-duration to 24h and collect-addr-util-interval to 1h, memory used by the DHCP server to maintain address utilization data is 24 times 68 bytes for each scope; that is, 10x24x68 or 16KB.

### **collect-performance-statistics** [bool](#) default = false

Controls whether the DHCP server collects statistics for performance monitoring.

### **collect-sample-counters** [bool](#) default = disabled

Controls whether the DHCP server collects activity statistics independently of the log-settings attribute flag. If false, this attribute is disabled.

If true, this attribute enables collecting activity statistics.

Note: These activity statistics counters are also enabled if you enable 'activity-summary' logging is enabled (see 'log-settings').

### **default-free-address-config** [nameref\(AddrTrapConfig\)](#)

Specifies the default SNMP free-address trap configuration for the server. All scopes that are not individually configured use this default free-address value.

### **defer-lease-extensions** [bool](#) default = true

Determines whether the DHCP server can defer extending leases to their full lease expiration time.

If enabled (true), the DHCP server can defer extending leases to their full expiration time. Rather than give the client the full lease time, the server instead gives the client the remaining time on the existing lease, improving server performance and throughput. The DHCP server typically defers extending leases if the client renews early; that is, before the client's renewal time (T1) elapses.

If enabled, the DHCP server can save committing the lease to disk and updating its failover partner with an extended expiration time.

If disabled, the DHCP server always attempts to extend leases to their full expiration time.

lease expiration time. However, there may be conditions that prevent the server from extending the lease for the configured lease time; for example, failover protocol re

### **delete-orphaned-leases** **bool** default = false

Controls how the DHCP server handles orphaned leases. Default is false (disabled).

A lease is orphaned if:

--It does not match a range or reservation in a configuration  
--If it appears in the lease state database with a VPN ID that does not match a VPN in the database.

If this attribute is true, the server deletes orphaned leases when the server next reloads. This ensures that the lease state database has no old, unconfigured leases in it, but it can cause the total loss of lease state information due to a configuration change.

If this attribute is false, the server ignores the entries in the lease state database. If the lease was orphaned due to a configuration mistake, then when you correct the mistake, the DHCP server can use the lease.

Whether true or false, the server cannot use the lease.

### **delete-orphaned-subnets** **bool** default = false

Controls how the DHCP server handles information about an orphaned subnet; that is, whether it keeps the entry in the lease state database or deletes it. Default is false (disabled).

As the DHCP server starts up, it tries to locate the parent VPN and DHCP address block of each DHCP subnet. If the server cannot locate a parent DHCP address block that contains the subnet, the server uses this attribute to decide.

### **dns-timeout** **int** default = 60000, required

Controls the number of milliseconds that the DHCP server waits for a response before retrying dynamic DNS requests.

### **docsis-version-id-missing** **string**

Sets the string value (maximum 255 characters) to substitute for the `%@docsis-vers%` variable in the policy command boot-file attribute. No default.

DOCSIS devices must provide their version ID (for example, "docsis1.0") in the vendor-class-ID option. The server substitutes the DOCSIS version ID as the value of the boot-file option. If the vendor-class-ID option is missing, or is not syntactically correct, the server uses the docsis-version-id-missing attribute for the boot-file option.

This substitution occurs if the DHCP request packet does not contain a vendor-class-ID option or the option does not contain a DOCSIS version ID.

### **drop-old-packets** **int** default = 4

Sets the time, in seconds, that a packet can age and still be processed.

The server attempts to read as many packets as possible from the UDP input queue, and then process them quickly. If the server is very busy, it can sometimes become flooded with packets. This could delay processing some packets.

In the DHCP protocol, however, some clients automatically retry packets that have not been processed in a few seconds--so allowing the server to process packets that are

older than a small number of seconds. This can increase congestion without providing any real value for the client. The drop-old-packets parameter is the number of seconds a packet can age and still be processed by the DHCP server. If a packet is more than the value of drop-old-packets when processed by the DHCP server, the server drops the

**drop-packet-on-extension-failure** `bool` default = enabled

Causes the server to drop a packet (if possible) when it encounters an extension failure. Default is true (enable)

**enhanced-sample-counters** `bool` default = enabled

Enables collection of enhanced statistics counters by the server, which are then available with DHCP server status. The enhanced counters provide more detailed information, but cost the server some performance to maintain. Currently, enables collecting milliseconds ACK/Reply latencies (in seconds based) and scope aggregation data (even if not explicitly configured).

**equal-priority-most-available** `bool` default = disabled

Controls address allocation among scopes in the same network. This attribute determines how the DHCP server allocates an address to a new client when multiple scopes have the same nonzero allocation priority. Default is false (disabled). If disabled, the server uses the scope with the fewest addresses to allocate an address to a new client (if not in a limitation list).

If enabled, the server uses the scope with the most available addresses to allocate an address to a new client (if not in a limitation list).

In either case, if a client is in a limitation list, among those scopes of the same priority, the one that contains other clients in the same list is always used.

**expression-configuration-trace-level** `int` default = 2

Sets the trace level to use when configuring DHCP expressions. Default is 2 (failure retry).

The range is from 0 through 6, with 0 being the lowest amount of tracing and 6 the highest:

- 0 No additional tracing
- 1 No additional tracing
- 2 Failure retry
- 3 Function definitions
- 4 Function arguments
- 5 Variable lookups and literal details
- 6 Everything

Setting a high expression-configuration-trace-level imposes a performance penalty, since expressions are configured on the server when you restart the server.

**expression-trace-level** `int` default = 2

Sets the trace level to use when executing DHCP expressions. Default is 2.

The range is from 0 through 6, with 0 being no tracing and 6 the highest amount of tracing:

- 0-No tracing
- 1-Failures, including those protected by (try ...)
- 2-Total failure retries (with trace level = 6 for retry)
- 3-Function calls and returns
- 4-Function arguments evaluated
- 5-Print function arguments

6-Datatype conversions (everything)  
Setting this attribute to any level other than 0, 1, or imposes a considerable performance penalty.  
The setting of 1 only traces when there is a failure in expression. The default setting of 2 re-executes evaluating an expression that fails at the outermost level with the expression-trace-level=10 for the duration of the re-execution, to provide maximum debugging assistance.

**extension-trace-level** `int` default = 0

Sets the value of the extension trace level for every request object. Default is 0.  
The range is from 0 through 3, with 0 being very little and 3 the highest amount of tracing.

**failover-bulking** `bool` default = true

With failover enabled, controls whether a failover BIND (BNDUPD) contains multiple lease state updates. Default (enable).  
This attribute affects only the lease state updates that client generates.

**failover-poll-interval** `time` default = 15s

With failover enabled, controls the space of time, in seconds that elapses between polls of failover partners to confirm network connectivity. Default is 15s.

**failover-poll-timeout** `time` default = 60s

With failover enabled, defines the space of time, in seconds after which a poll between partners times out. Default is 60s.  
This attribute signals that the partners cannot communicate due to lost network connectivity. The partners then change operational states appropriately.

**failover-recover** `date` default = none

With failover enabled, sets the time at which the server is initialized and goes into RECOVER state. No default.  
If failover-recover is non-zero, and the server has no record of a previous failover state in its stable storage then the server assumes that it was previously operational but lost all of its stable storage.  
The time value in the failover-recover parameter represents a time equal to or later than the last known time the server could have been operational. The server attempts to refresh its information from its failover partner by entering RECOVER state and requesting a complete update of all binding information.

Time values for this attribute can be in hours (for example -2h two hours ago) or month day hour:minute[:second] year. Use the local time of the nrcmd process.

Formats for the date are:

-

where `is` is a decimal number and `is` one of 's', 'm', 'h', 'd', 'w', in which 's' is seconds, 'm' is minutes, 'h' is hours, 'd' is days and 'w' is weeks.

[::]

where `is` is the name or first three letters of the name of the month, `is` is the hour on a 24-hour clock, and `is` is the fully-specified year or a two-digit representation of the year (98 = 1998, 99 = 1999 and all other two digit values XX).

**force-dns-updates** `bool` default = false

Controls whether the DHCP server retries a DNS update whenever a client renews its lease, even if it appears to the server that the update was already completed successfully. Default is false (disable).

This attribute uses one of the following values:

- forward DnsUpdateConfig object (if configured)

- reverse DnsUpdateConfig object (if configured)

- the default (or where appropriate the server configuration value or default value).

### **get-subnet-mask-from-policy** `bool` default = false

Controls whether the DHCP server searches all relevant policies for a subnet mask option to construct a response to a client. Default is false (disable).

Normally, the DHCP server retains the subnet mask configuration in the scope containing the base being granted to the DHCP client.

### **ha-dns-failover-timeout** `rangeint(30-120)` default = 30

Sets the maximum time period in seconds that the DHCP server waits for replies from a DNS server before failing over to a partner. Default is 30s.

To use this attribute, first configure your DHCP server to perform HA DNS updates.

If DHCP is configured for HA-DNS updates, `dns-timeout` parameters are set to fit `max-dns-retries` within `ha-dns-failover-timeout`.

For example, with a default `ha-dns-failover-timeout` value of 30 seconds and `max-dns-retries` of 3, `dns-timeout` is set to 6 seconds, so that after 3 retry attempts and an

`ha-dns-failover-timeout`, the DHCP server fails over. The server uses a minimum limit value for a `dns-timeout` of 2

and `dns-retries` of 1.

### **hardware-unicast** `bool` default = enabled

Controls whether the DHCP server sends unicast rather than broadcast responses when a client indicates that it accepts a unicast.

Note: This attribute is only available on these operating systems: Solaris, Windows 2000, and Windows NT

### **ignore-cisco-options** `string`

Skips processing the Cisco-specific DHCP options specified by name in the comma-separated list. No default.

Allowable option names are `vpn-id` (185), `cisco-vpn-id` (2) and `subnet-alloc` (220). Use this attribute only if

clients use the options for other purposes.

### **ignore-icmp-errors** `bool` default = enabled

Controls how the DHCP server handles ICMP ECHO (ping-beef offer) requests. Default is true (enable).

If you enable this attribute and configure the DHCP server to send ICMP ECHO requests, the server makes unavailable

addresses for which it receives an ECHO reply within its configured timeout period.

If you disable this attribute, the DHCP server also treats ICMP DEST\_UNREACHABLE and TTL\_EXPIRED error messages that

it receives after sending ICMP ECHO requests as grounds for an address unavailable.

### **ignore-requests-for-other-servers** `bool`

Controls whether to prevent the normal DHCP server response to client requests for other servers. No

default.

Normally, if the DHCP server detects a client requesting from another server for an address that this server is configured to control, it sets the lease to unavailable. Clients, however, might send request packets with bad options--rather than packets actually directed to other servers--that the server wrongly interprets as an unavail address. Enabling this attribute prevents this.

**import-mode** **bool** default = disabled

Controls whether the DHCP server recognizes only packets generated from the import leases command, ignoring all others. Default is false (disable).

Use this attribute to update your DHCP server, preventing clients from receiving addresses during this period.

**inhibit-busy-optimization** **bool** default = false

Controls whether the server uses optimization to recover periods of congestion. Default false (disable).

By default, the DHCP server determines that it is heavily congested when the number of request packets reaches two-thirds of total allocated. The server logs a message and attempts to recover from the congestion by performing several optimizations. For example, it relaxes the requirement to keep the client's last transaction time updated to the granularity specified by the last-transaction-time-granularity attribute.

When the number of request packets drops to one-third of total allocated, the server logs a message and returns to normal operation. If you enable the inhibit-busy-optimization attribute, the server does not use the optimizations or send messages when it gets congested.

**initial-environment-dictionary** **string**

Contains attribute-value pairs that initialize all environment dictionaries used within the DHCP server. Use these attribute pairs to configure extensions or expressions without having to re-write the executable code in the extension or expression. The string must have the format:

"attribute1=value1,attribute2=value2, ... ,attributen=value

**ip-history** **bool** default = disabled

Enables the lease history database. Default is false (disabled).

**ip-history-detail** **bool** default = false

Controls whether to record detailed data for the IP history database. Default is false (disabled).

**ip-history-max-age** **time** default = 4w

If ip-history is enabled, determines how long IP records are kept in the database. Default is 4w.

The server accumulates database records over time as lease bindings change. The ip-history-max-age attribute establishes a limit on the age of the history records kept in the database. The server periodically examines the lease history records and establishes an age threshold based on this parameter, and deletes any records that represent bindings that end before the threshold. The history records are trimmed by default a day, at 3:00 a.m. local time.

**last-transaction-time-granularity** **time** default = 60s

Sets the time, in seconds, to guarantee that the last transaction time is accurate. Default is 60s.

Do not set this lower than the default of 60 seconds. For

optimal performance, set it to a value that is greater than that of your lease interval.

By default the server maintains an accurate record of the time when it last interacted with a DHCP client concerning a lease. This setting provides a control on how accurate this record is guaranteed to be. A setting of 300 seconds, for instance, would allow the server to avoid database updates whose sole purpose is to update a last transaction time that is less than 5 minutes in the past.

### **ldap-mode enumint(round-robin=1, failover=2)**

Determines the preference for using LDAP servers if multiple LDAP servers are configured. No default.

This attribute has two possible values:

- 1 round-robin-The DHCP server ignores LDAP server preferences. It treats all LDAP servers (those configured to handle client queries and those configured to handle lease-state updates) equally.
- 2 failover-The DHCP server uses the active LDAP server with the lowest preference. If the preferred server loses its connection or fails, the DHCP server uses the next server in preference order. The DHCP server uses servers with equal preference in round-robin order.

### **log-settings flags(default=1, incoming-packets=2, missing-options=3, incoming-packet-detail=4, outgoing-packet-detail=5, unknown-criteria=6, dns-update-detail=7, client-criteria-processing=9, failover-detail=10, ldap-query-detail=11, ldap-update-detail=12, ldap-create-detail=13, leasequery=14, dropped-waiting-packets=15, no-successful-messages=16, no-dropped-dhcp-packets=17, no-dropped-bootp-packets=18, activity=19, activity-summary=20, no-invalid-packets=21, no-reduce-logging-with-no-timeouts=23, minimal-config-info=24, no-failover-conflict=25, atul-detail=26, outgoing-packet-detail=27) default = default,incoming-packets,missing-options**

Determines which events to log in the log files. Default is default, incoming-packets, and missing-options.

Logging additional detail about events can help analyze problems. However, leaving detailed logging enabled for a long period can fill up the log files.

Possible flags are:

default

The default gives a low level of logging in several areas of the DHCP server. If you unconfigure the default, even minimal logging will not appear.

missing-options

This setting (on by default) will cause a message to be logged whenever an option requested by a DHCP client has not been configured in a policy and therefore cannot be supplied by the DHCP server.

incoming-packets

This setting (on by default) will cause a single line to be logged for every incoming packet. This is especially useful when initially configuring a DHCP server or a relay, in that an immediate positive indication exists that the DHCP server is receiving packets.

incoming-packet-detail

This setting will cause the contents of every DHCP packet received by the DHCP server to be interpreted in a human-readable way and printed in the log file. This enables the built-in DHCP packet sniffer for input packets. The log will fill up (and turn over) very rapidly when this

enabled. This setting also causes a significant per impact on the DHCP server and should not be left enabled as a matter of course.

#### outgoing-packet-detail

This setting will cause the contents of every DHCP packet transmitted by the DHCP server to be interpreted in a readable way and printed in the log file. This enables the built-in DHCP packet sniffer for output packets. The buffer will fill up (and turn over) very rapidly when this setting is enabled. This setting also causes a significant per impact on the DHCP server and should not be left enabled as a matter of course.

#### client-detail

This setting will cause a single line to be logged at the conclusion of every client-class client lookup operation. Each line will show the composite of the data found for that client as well as the data that was found in the client's configuration. It is useful when setting up a client-class configuration for debugging problems in client-class processing.

#### client-criteria-processing

This setting will cause a log message to be output whenever a scope is examined to find an available lease or when a scope is examined to determine if a lease is still available for a client who already has one. It can be very useful when configuring or debugging client-class scope criteria processing. It causes a moderate amount of information to be logged and should not be left enabled as a matter of course.

#### unknown-criteria

This setting will cause a single line log message to be output whenever a client entry is found which specifies selection criteria that is not found in any scope appropriate to the client's current network location.

#### dns-update-detail

This setting causes the server to log a message as it updates each dns update and as it receives replies to update requests.

#### ldap-query-detail

This setting will cause log messages to appear whenever the dhcp server initiates a query to LDAP server, receives a response and retrieves result or error messages.

#### ldap-update-detail

This setting will cause log messages to appear whenever the dhcp server initiates an update lease state to LDAP server, receives response and retrieves result or error messages.

#### ldap-create-detail

This setting will cause log messages to appear whenever the dhcp server initiates an lease state entry create to LDAP server, receives response and retrieves result or error messages.

#### leasequery

This setting will cause log messages to appear whenever lease query packets are processed without internal errors and receive an ACK or a NAK.

#### dropped-waiting-packets

If the value of max-waiting-packets is non-zero packets will be dropped if the queue length for any IP address exceeds the value of max-waiting-packets. If dropped-waiting-packets is set, the server will log a message whenever it drops a waiting packet from the queue for an IP address.

#### no-success-messages

This setting will cause the single line message that normally logged for every successful outgoing DHCP r packet to not appear. It affects logging only for s outgoing DHCP response packets.

no-dropped-dhcp-packets  
This setting will cause a single line message normal for every DHCP packet that is dropped due to DHCP configuration to not appear. (See no-invalid-packet messages associated with packets dropped because the invalid.)

no-dropped-bootp-packets  
This setting will cause the single line message norm logged for every BOOTP packet that is dropped to not

no-failover-activity  
This setting will cause normal activity and some war messages logged for failover to not appear. Serious log messages will continue to appear independent of log-setting.

activity-summary  
This setting will cause a summary message to appear minutes. It is useful when many of the no-xxx log s are enabled, to give some idea of the activity in th without imposing the load required for a log message corresponding to each DHCP message. The time period these messages can be configured with the DHCP serve property activity-summary-interval.

no-invalid-packets  
This setting will cause a single line message normal for every DHCP packet that is dropped due being inva not appear. (See no-dropped-dhcp-packets for messag associated with packets dropped due to DHCP server configuration.)

no-reduce-logging-when-busy  
Normally, the DHCP server will reduce logging when i very busy (i.e., when it has used over 2/3 of the av receive buffers (itself a configurable value)). It no-success-messages, no-dropped-dhcp-packets, no-dropped-bootp-packets, no-failover-activity, no-invalid-packets, and clear everything else except activity-summary. If no-reduce-logging-activity is the server will not do this. It will restore the pr settings when the server becomes unbusy (i.e., when used only 1/3 of the available receive buffers).

no-timeouts  
This setting will cause messages associated with tim of leases or offers not to appear in the log file.

minimal-config-info  
This setting will reduce the number of configuration printed when the server starts or reloads. In parti it will not log a message for every scope.

no-failover-conflict  
This setting will cause conflicts between failover p to not be logged.

atul-detail  
This setting causes the server to log messages when messages are received and processed.

v6-lease-detail  
This setting causes the server to log individual mes regarding DHCPv6 leasing activity (in addition to or

place of a single message per client transaction depends on no-success-messages, or client timeout event depends on no-timeouts).

### **mac-address-only** `bool` default = disabled

Controls whether the DHCP server uses the client's MAC address as the only client identifier. The standard behavior as specified in RFC 2132, is to use the client-id option (if present) as the unique client identifier. Default is disabled.

CAUTION: Use this attribute with care. When enabled, it precludes a MAC address from getting multiple IP addresses on the network. It forces the server to use a Client-Identifier created from the MAC address instead of the RFC described client-id contained in the request. This can preclude network devices that take multiple IP addresses. Enabling, or later disabling, this attribute can also have an operational impact. Clients that originally obtain addresses through a client-id cannot renew them once they are assigned addresses based on a MAC address.

### **map-radius-class** `enumint`(none=0, map-as-tag=1, map-as-class=2, append-to-tags=3) default = 0

Controls how to map the radius attribute, if present, in the client request relay-agent option:

0	none	Ignores the radius class name definition
1	map-as-tag	Maps the radius class to selection-tags
2	map-as-class	Maps the radius class directly to client-class name
3	append-to-tags	Appends the radius class to the selection-tags

### **map-radius-pool-name** `enumint`(none=0, map-as-tag=1, map-as-class=2, append-to-tags=3) default = 0

Controls use of the radius framed-pool attribute, if present, in a client relay-agent option.

0	none	Ignores the framed-pool attribute
1	map-as-tag	Maps the framed-pool attribute to selection-tags
2	map-as-class	Maps framed-pool attribute directly to client-class name
3	append-to-tags	Appends the user-class-id to the selection-tags

### **map-user-class-id** `enumint`(none=0, map-as-tag=1, map-as-class=2, append-to-tags=3) default = 0

Controls how the server uses the user-class-id option.

0	none	Ignores the user class-id (default)
1	map-as-tag	Maps the user-class-id to selection-tags
2	map-as-class	Maps user-class-id directly to a client-class name
3	append-to-tags	Appends the user-class-id to the selection-tags

### **max-client-leases** `rangeint`(1-65535) default = 50

Sets the maximum number of leases, regardless of state reserved or not, that the server can associate with a DHCP client. A DHCPv6 lease is always associated with a client. If the client is not, it is deleted.

This setting is to prevent a client from using lots of leases (such as by issuing many requests with different IAID values).

is not intended to limit the number of active leases a c have.

While 1 can be configured as the minimum for backwards compatibility, the server uses a minimum of 2 to allow for a client that usually only has a single lease to be renamed or moved to a different prefix on the link.

This limit is not applied when existing leases are loaded into the lease state database during server start-up.

### **max-dhcp-requests** `int` default = 500, required

Controls the number of buffers that the DHCP server allocates for receiving client requests.

When you enable failover, allocate at least 150 buffers. 1500 buffers might be reasonable for high capacity installations.

Caution: Increasing the number of buffers can degrade performance. Cisco recommends using the default value in most situations. When buffer size exceeds capacity, a backlog of DHCP activity can clog the server with requests that become stale before they are processed. This increases the processor load and might severely degrade performance as clients try to obtain a new lease. A lower buffer setting throttles requests and avoids wasted processing on requests that would otherwise be stale.

When using LDAP client lookups, buffers should not exceed the LDAP lookup queue size defined by the total number of LDAP connections and the maximum number of requests allowed per connection. Set the LDAP queue size to match the LDAP server capacity to service client lookups.

### **max-dhcp-responses** `int` default = 1000, required

Controls the number of buffers that the DHCP server allocates for responses to client requests. The server ignores this value if max-dhcp-requests is higher, or other configuration options such as failover require that the value be set higher than configured.

### **max-dns-renaming-retries** `int` default = 3, required

Controls the number of times that the DHCP server can attempt adding a host into DNS, even if the DHCP server detects the hostname is already present in DNS. The DHCP server attempts to modify the hostname in order to resolve a conflict on a failed update.

### **max-dns-retries** `int` default = 3, required

Controls the number of times that the DHCP server can attempt to send dynamic updates to a DNS server.

### **max-dns-ttl** `int` default = 86400, required

Sets the time to live (TTL) ceiling, in seconds, for DNS records added through dynamic updates. When the DHCP server adds a record, it uses a TTL of the minimum of either this ceiling or one-third the lease time.

### **max-ping-packets** `int` default = 500

Sets the number of buffers the server allocates for sending and receiving ICMP ping messages. See the scope 'ping-client' command.

### **max-waiting-packets** `int` default = 6

Sets the maximum number of packets that can wait for a particular IP address.

The server queues only the most recently received n packets.

(of an address) for processing. If an additional packet associated with that address arrives and n packets are queued, the server drops the oldest packet and queues the new one. It also drops duplicate packets (whose XID, client MAC address are the same as one already queued). A value of 0 indicates that there is no maximum queue length and all packets are processed. Dropped packets are logged if the log setting `dropped-waiting-packets` is set. It is off by default.

### **mcd-blobs-per-bulk-read** `rangeint(1-2500)`

Determines the number of binary large objects (blobs) in read. Default 256. Range, 1 to 2500. Use this attribute to tune DHCP start and reload times. Generally, a higher value will result in faster server reload time, at the cost of using more memory.

### **min-dns-ttl** `time default = 10m`

Minimum value for time to live (TTL) in seconds, for DNS records added through dynamic updates. When the DHCP server adds a new record, the TTL value will be `min-dns-ttl` if one third the current TTL time is less than the `min-dns-ttl` value.

### **multicast-addresses** `ip6addr default = ff02::1:2,ff05::1:3`

Controls the default multicast addresses that are enabled on interfaces. The address `ff02::1:2` is required if any DHCPv6 clients are directly connected to the link associated with an interface. The address `ff05::1:3` is the default multicast address that relay agents use to relay DHCPv6 requests.

### **one-lease-per-client** `enumint(disabled=0, last-client-preferred=1, first-client-preferred=2) default = disabled`

Controls whether the DHCP server releases other leases a client might have on other LAN segments on this server.

- 0 disabled
- 1 last-client-preferred
- 2 first-client-preferred

Within one LAN segment, the DHCP server never allocates more than one DHCPv4 address to a single client. Across multiple independent network LAN segments, however, a single client might have one address allocated on several networks.

In an enterprise environment this might happen when a user travels from building to building, causing no particular problems. In a service provider context, this might happen if an unapproved user attempts to clone the MAC address of a legitimate subscriber, causing a theft of service.

The one-lease-per-client feature limits a single client's lease over all of the networks configured on the DHCP server. This limit only affects a client's ability to have concurrent leases to different IP addresses on more than one network at a time on the DHCP server.

In the last-client-preferred approach, the client with the most recent lease is given preference, and any other leases held by the same client are released.

In the first-client-preferred approach, the first lease a particular client receives from the DHCP server is the only lease that client is allowed. In the event that the client moves, the first lease must be made available (perhaps by the `force-available` command or by letting it expire) before the client is allowed to lease another IP address on a different

network.

Caution: Use the first-client-preferred approach with gr care, since manual intervention might be required to ens proper operation.

Both forms of one-lease-per-client require additional bc overhead beyond that normally done in the DHCP server.

### **priority-address-allocation** **bool** default = disabled

Controls address allocation among scopes in the same net within an individual scope as well.

When enabled, any scope without an explicit setting for allocation-priority is configured with an allocation-pri equal to the network number of the scope. Similarly, an without an explicit setting for allocate-first-available is considered enabled.

Explicit settings for either of these scope attributes c the priority-address-allocation set for that scope.

This attribute gives the administrator a way to change a allocation in a server wide manner without having to sep configure each scope.

### **return-client-fqdn-if-asked** **bool** default = true

Controls whether the system returns the client FQDN opti in the outgoing packet to the client if the incoming pac contains the FQDN option. Default false (disable).

If true (enable), the option flags are always set to 0x3 and the RCODE1 and RCODE2 to 255. Any string contained i incoming packet is returned, even if the use-client-fqdn attribute is disabled and regardless of the actual FQDN.

### **save-vendor-class-id** **bool** default = false

Determines whether to store the vendor=class-id, as furnished in a dhcp request option 60, as part of the le record in either ldap, mcd or both.

### **skip-client-lookup** **bool** default = false

Determines whether the DHCP server looks up the client e in the database to do client-class processing. This valu can be examined as well as changed in a script at the pre-client-lookup extension point. Default false (disable).

If true, the server skips the client entry.

### **sms-lease-interval** **int** default = 1100

Sets the amount of time, in milliseconds, that the DHCP waits between sending addresses to the System Management (SMS) when executing updateSms command.

### **sms-library-path** **string**

Overrides the internal default value for the SMS.dll. D is an empty string.

If the string is empty, the system defaults to the inter server default of smsrsgen.dll.

If the string is not empty, its value overrides the inte SMS library name, smsrsgen.dll. If the system path does the location of the SMS dll, you should provide the absc of the dll.

### **sms-network-discovery** **int** default = 0

Determines whether the DHCP server generates SMS network discovery records.

If this attribute is set to 0, you disable SMS network discovery. If it is set to 1, you enable discovery.

Use this attribute in conjunction with the dhcp updateSM command.

### **sms-site-code** [string](#)

Specifies the site code name of the SMS server that receives discovery records when you use the updateSMS keyword. For proper functioning, make sure that you initialize the attribute to the appropriate site code. The default value is an empty string, but this prohibits discovery to complete successfully. So, you must provide site code.

### **synthesize-reverse-zone** [bool](#) default = enabled

Controls whether the DHCP server automatically generates name of the reverse zone (in-addr.arpa or ip6.arpa) that PTR records updates. Default is true (enable). If true, you are not required to configure an explicit dns-reverse-zone-name in the DNS update configuration. The DHCP server uses the IP address of each lease and the dn attribute of the scope or the reverse-zone-prefix-length of the prefix to synthesize the reverse zone name for each. The server trims the specified host bytes from the low-order bytes of the address, and turns the remaining bytes into name of the form '.b.a.in-addr.arpa.' The host-bytes low-order bytes of the address are used to form the host within that zone. Similarly, the prefix length is used to form the 'ip6.arpa' zone name, and the low-order bytes are used to form the hostname.

### **traps-enabled** [flags](#)(all=1, server-start=2, server-stop=3, free-address-low=4, high=5, dns-queue-size=6, other-server-down=7, other-server-up=8, duplicate-address-conflict=10, failover-config-error=11, free-address6-low=12, free-address6-high=13, duplicate-address6=14, duplicate-prefix6=15)

Determines the traps that this server is configured to send.

- 1 all  
Sends notifications for all server events.
- 2 server-start  
Sends notifications whenever the server is started or reinitialized.
- 3 server-stop  
Sends notifications whenever the server is stopped.
- 4 free-address-low  
Sends notifications when the number of free IP addresses becomes less than or equal to the low threshold.
- 5 free-address-high  
Sends notifications when the number of free IP addresses exceeds the high threshold after having previously triggered the free-address-low trap.
- 6 dns-queue-size  
Sends notifications when the DHCP server's DNS queue fills and the DHCP server stops processing requests.
- 7 other-server-down  
Sends notifications when another server (DHCP, DNS or LDAP) stops responding to this DHCP server.
- 8 other-server-up  
Sends notifications when another server (DHCP, DNS or LDAP) responds after having been unresponsive.
- 9 duplicate-address  
Sends notifications whenever a duplicate IP address is detected.

- 10 address-conflict  
Sends notifications when an address conflict with DHCP server is detected.
- 11 failover-config-error  
Notifies when a configuration mismatch between DHCP failover partners occurs.
- 12 free-address-low  
Sends notifications when the number of free IPv6 addresses becomes less than or equal to the low threshold.
- 13 free-address-high  
Sends notifications when the number of free IPv6 addresses exceeds the high threshold after having previously triggered the free-address-low trap.
- 14 duplicate-address6  
Sends notifications whenever a duplicate IPv6 address is detected.
- 15 duplicate-prefix6  
Sends notifications whenever a duplicate prefix is detected.

No default.

#### **trim-host-name** `bool` default = enabled

Controls whether or not DHCP server trims the hostname to the first period (or dot) character (used to update DNS records and to return host-name-option to clients). Default is true (enable).

If true, the hostname is truncated before the period.

If false, the DHCP server retains the period characters in the hostname.

#### **update-dns-for-bootp** `bool` default = enabled

If the server is replying to a BOOTP request, and is off-lease from a Scope which is configured to perform DNS updates, will check this property before beginning the DNS update feature. This property allows an administrator to prevent DNS updates for clients, while allowing updates for DHCP clients.

#### **upgrade-unavailable-timeout** `time` default = 24h

Controls the time given to a lease in the database that has expired; that is, it became unavailable prior to installing Network Registrar. The DHCP server uses the upgrade-unavailable-timeout for the expiration time of a lease. Default is 86400 seconds (1 day).

#### **use-client-fqdn** `bool` default = true

Controls whether the server examines the client-FQDN option for the hostname. Default is true (enable).

If true, the server ignores any characters after the first period because the domain is determined from the scope.

If false, the server does not determine the hostname using the client-FQDN option. This is useful if the client is sending unexpected or junk characters.

#### **use-client-fqdn-first** `bool` default = true

Controls whether the DHCP server looks at the client-FQDN option on incoming packets first, before looking at the 'hostname' option. Default is true (enable).

If true and the client-FQDN option specifies a hostname, the server uses that hostname.

If the client-FQDN option is not present in the incoming packet, the server uses the hostname from the 'hostname' option. If false, the server also uses the hostname from the

'hostname' option.

### **use-dns-update-prereqs** **bool** default = true

Controls whether the DHCP server adds prerequisites to DNS update messages. Default true (enable).

If true, the DHCP server includes prerequisites in DNS update messages to make sure the client is using the domain name before it is updated with the current lease (IP address).

If false, the DHCP server assumes the requesting client is entitled to the domain name. In this case, it does not include prerequisites in the DNS update message, and associates the lease with that domain name.

Note: The DHCP server always adds prerequisites to the DNS update message while adding a new domain name record on behalf of a client acquiring a new lease and removing the domain record when a client releases its lease or the lease expires.

### **use-host-name** **bool** default = true

Specifies whether the server looks at the 'hostname' option for the hostname. Default is true (enable).

If true, the server obtains the hostname from the 'hostname' option.

If false, the server does not obtain the hostname from the 'hostname' option. This is useful if the client is sending unexpected junk characters.

### **use-ldap-client-data** **bool** default = disabled

Controls whether the DHCP server attempts to read client data using the configuration supplied by the 'ldap' command. Default is false (disabled).

### **v6-client-class-lookup-id** **expr**

Defines the expression used to assign a client-class based solely on data contained in an incoming DHCPv6 client request. No default.

The expression must return a string that is the name of a currently configured client-class; otherwise, the expression must return the string ''. Any return that is not a string containing the name of a currently configured client-class is considered an error.

### **v6-default-free-address-config** **nameref**(AddrTrapConfig)

Specifies the default SNMP v6 free-address trap configuration for the server. All Prefixes and Links that are not individually configured use this default free-address value.

### **validate-client-name-as-mac** **bool** default = false

Controls whether the Network Registrar user interfaces require that the name of client entries is a valid MAC address (or a literal string 'default'). Default is false (disabled).

If true, the user interfaces convert the name to the canonical MAC address format: 1,6,xx:xx:xx:xx:xx:xx. The DHCP server uses this as the default client entry lookup key.

If false, the user interfaces allow creating client entries with arbitrary names, which could match the lookup keys generated from the client-lookup-id expression.

### **vpn-communication** **bool** default = true

Controls the ability of the DHCP server to communicate with clients that are on a different VPN from the server. Default is true (enable).

If true, the server communicates with DHCP clients residing on the same VPN as the server.

a different VPN by using an enhanced DHCP Relay Agent capability. This enhanced DHCP Relay Agent capability is indicated by the appearance of the server-id-override sub-option in the relay agent information option (Option 82).

## dhcp-address-block

`dhcp-address-block` - Defines a contiguous range of IP address space from which the DHCP server may allocate subnets

### Synopsis

```
dhcp-address-block <name> create <address> [<attribute>=<value> ...]
dhcp-address-block <name> delete
dhcp-address-block list
dhcp-address-block listnames
dhcp-address-block <name> show
dhcp-address-block <name> listsubnets

dhcp-address-block <name> get <attribute>
dhcp-address-block <name> set<attribute>=<value> [<attribute>=<value> ...]
dhcp-address-block <name> unset <attribute>
dhcp-address-block <name> enable <attribute>
dhcp-address-block <name> disable <attribute>
```

### Description

The `dhcp-address-block` command creates and sets attributes for Network Registrar DHCP address blocks. The command applies only to address block objects that are designated in the DHCP server for subnet allocation to clients. When a DHCP server receives a request to allocate a subnet to a client, it does so by subdividing its available address-blocks. In this context, a DHCP address block is a contiguous range of IP address space that is delegated to the DHCP server for assignment. The DHCP server expects to subdivide these DHCP address blocks for delegation to some other server or device, or for its own use in interaction with DHCP clients.

DHCP address blocks can parent one or more subnets. Subnets are also contiguous ranges of IP address space that are bound to a specific client, usually a router or another DHCP server. DHCP address blocks and subnets are similar to scopes in that they contain address ranges and other attributes necessary to configure the DHCP client-server interaction. Unlike scopes, DHCP address blocks and subnets do not have address ranges available for assignment to DHCP clients and do not contain reserved addresses.

In a virtual private network (VPN) deployment where multiple VPNs use the same private address space, you can use logically identical DHCP address blocks simultaneously on multiple VPNs.

### Examples

```
nrcmd> dhcp-address-block example1 create 10.10.0.0/16
nrcmd> dhcp-address-block example1 set policy=p2
nrcmd> dhcp-address-block example1 show
```

### Status

## See Also

[dhcp-subnet](#)

```
vpn [] (AT_STRING, Optional, default: <none>)  
This is a "virtual" property. Use this property to set or  
get the vpn-id by vpn name instead of by id.
```

## Attributes

### address [subnet](#) required

Determines the IP subnet address (in the address/mask a DHCP address block. This value is defined when you address block.

### default-subnet-size [int](#) default = 28

Sets the default subnet size for allocations from this block.

### deprecated [bool](#) default = false

Determines whether the server deactivates a DHCP address. The server ignores a deprecated DHCP address block for allocations. It allows existing clients to renew their indicates to them that the subnet is deprecated. The prepares to release the deprecated subnet or subnets server.

### embedded-policy [obj\(0\)](#)

Displays the embedded policy object for this DHCP address. Read-only. Use the dhcp-address-block-policy command embedded policy.

### name [string](#)

Defines the name of the address-block. This value is when you create the address block.

### policy [nameref\(Policy\)](#) default = default, required

Identifies the name of the policy associated with this

### segment-name [string](#)

Designates the LAN segment name for this DHCP address multiple, logical IP subnets on a single, physical network. DHCP address block the same segment-name string. The character case when comparing values.

### selection-tags [string](#)

Lists tag strings that are compared with incoming selection an allocation request. All tags in the request must match address block's selection tags for that block to satisfy. Separate multiple tags with a comma (do not include character names).

### vpn-id [int](#) default = 0

Sets the VPN identifier for the VPN that contains this

## dhcp-address-block-policy

```
dhcp-address-block-policy - Edits a DHCP policy embedded in an  
address-block
```

## Synopsis

```
dhcp-address-block-policy <name> delete
```

```

dhcp-address-block-policy <name> set <attribute>=<value>
                                [<attribute>=<value> ...]
dhcp-address-block-policy <name> get <attribute>
dhcp-address-block-policy <name> disable <attribute>
dhcp-address-block-policy <name> enable <attribute>
dhcp-address-block-policy <name> show

dhcp-address-block-policy <name> setLeaseTime<time-val>
dhcp-address-block-policy <name> getLeaseTime

dhcp-address-block-policy <name> setOption<opt-name | id> <value>
dhcp-address-block-policy <name> getOption<opt-name | id>
dhcp-address-block-policy <name> unsetOption<opt-name | id>
dhcp-address-block-policy <name> listOptions
dhcp-address-block-policy <name>
    setVendorOption <opt-name | id> <opt-set-name> <value>
dhcp-address-block-policy <name>
    getVendorOption <opt-name | id> <opt-set-name>
dhcp-address-block-policy <name>
    unsetVendorOption <opt-name | id> <opt-set-name>
dhcp-address-block-policy <name> listVendorOptions

```

## Description

The `dhcp-address-block-policy` command lets you configure a DHCP policy embedded in a DHCP address block. An embedded policy is a collection of DHCP option values and settings associated with (and named by) another object -- in this case an address block. An `dhcp-address-block-policy` is created implicitly when you first reference it, and is deleted when the address block is deleted.

You can set individual option values with the `setOption` command, unset option values with the `unsetOption` command, and view option values with the `getOption` and `listOptions` commands. When you set an option value the DHCP server will replace any existing value or create a new one as needed for the given option name.

## Examples

## Status

## See Also

[policy](#), [client-policy](#), [client-class-policy](#), [scope-policy](#)

## Attributes

### **affinity-period time**

Associates a lease in the AVAILABLE state with the client last held the lease. If the client requests a lease during affinity period, it is granted the same lease; that is, renewals are prohibited, then it is explicitly not given. Because of the vast IPv6 address space and depending on generation technique, it could be millions of years before address ever needs reassignment to a different client, a is no reason to hold on to this information for that long.

To prohibit renewals enable either the `inhibit-all-renew` or the `inhibit-renews-at-reboot` attribute.

**allow-client-a-record-update** `bool` default = disabled

Determines if a client is allowed to update A records. If the client sets the flags in the FQDN option to indicate it wants to do the A record update in the request, and its value is TRUE, the server allows the client to do the A update; otherwise, based on other server configurations, does the A record update.

**allow-dual-zone-dns-update** `bool` default = disabled

Enables DHCP clients to perform DNS updates into two DNS zones. To support these clients, you can configure the DHCP server to allow the client to perform an update, but also to perform an update on the client's behalf.

**allow-lease-time-override** `bool` default = disabled

Gives the server control over the lease period. Although a client can request a specific lease time, the server need not honor the request if this attribute is set to false (the default). Even if set to true, clients can request only lease time shorter than those configured for the server.

**allow-non-temporary-addresses** `bool` default = true

Determines whether DHCPv6 clients can request non-temporary (IA\_NA) addresses.

The default is to allow clients to request non-temporary

**allow-rapid-commit** `bool` default = false

Determines whether DHCPv6 clients can use a Solicit with Rapid Commit option to obtain configuration information with fewer messages. To permit this, make sure that a single server is servicing clients.

This attribute has special handling during the policy hierarchy processing when checking the Prefix policies (embedded or for the Prefixes on a Link. The Prefixes for the Link are processed in alphabetic (case blind) order. For each Prefix embedded and then named policy are checked. Only Prefixes the client has access (based on selection tags, etc.) are checked.

- If any of the prefix policies has this attribute set FALSE, Rapid Commit is not allowed.
- If at least one has it set to TRUE, Rapid Commit is allowed.
- Otherwise, the remaining policies in the hierarchy are checked.

The default is not to allow clients to use Rapid Commit.

**allow-temporary-addresses** `bool` default = true

Determines whether DHCPv6 clients can request temporary addresses.

The default is to allow clients to request temporary add

**default-prefix-length** `rangeint(0-128)` default = 64

For delegation, specifies the default length of the delegation if a router (client) does not explicitly request it.

The default length must always be less than or equal to the length of the prefix range.

**forward-dnsupdate** `nameref(DnsUpdateConfig)`

Specifies the name of the update configuration that determines which forward zones to include in updates.

**forward-zone-name** `dname`

Designates an optional forward zone for DNS updates.

**giaddr-as-server-id** **bool** default = false

Enables the DHCP server to set the server-id option on a and a DHCPACK to the giaddr of the incoming packet, inst IP address of the server (the default action). This causes all unicast renews to be sent to the relay a of directly to the DHCP server, and so renews arrive at server with option-82 information appended to the packet. Some relay agents may not support this capability and, i complex configurations, the giaddr might not actually be to which the DHCP client can send A unicast packet. In t the DHCP client cannot renew a lease, and must always pe rebind operation (where the DHCP client broadcasts a req of sending a unicast to what it believes is the DHCP ser

**grace-period** **time** default = 5m

Defines the length of time between the expiration of a l and the time it is made available for reassignment.

**inhibit-all-renews** **bool** default = false

Causes the server to reject all renewal requests, forcin to obtain a different address any time it contacts the D

**inhibit-renews-at-reboot** **bool** default = false

Permits clients to renew their leases, but the server fo them to obtain new addresses each time they reboot.

**limitation-count** **int**

Specifies the maximum number of clients with the same li that are allowed to have currently active and valid leas

**offer-timeout** **time** default = 2m

Instructs the server to wait a specified amount of time has offered a lease to a client, but the offer is not ye At the end of the specified time interval, the server ma lease available again.

**packet-file-name** **string**

Identifies the boot-file to use in the boot process of a The server returns this file name in the 'file' field of The packet-file-name cannot be longer than 128 character

**packet-server-name** **string**

Identifies the host-name of the server to use in a clien process. The server returns this file name in the 'sname of its replies. The packet-server-name field cannot be l than 64 characters.

**packet-siaddr** **ipaddr**

Identifies the IP address of the next server in the clie process. For example, this might be the address of a TFT used by BOOTP clients. The server returns this address i 'siaddr' field of its replies.

**permanent-leases** **bool** default = disabled

Indicates whether leases using this policy are permanent to requesting clients. If leases are permanently granted ignores the configured dhcp-lease-time option value and maximum lease time.

**preferred-lifetime** **time** default = 1w

Assigns the default and maximum preferred lifetime for l DHCPv6 client interfaces. Expressed in seconds and relat the time the server sent the packet, this attribute sets length of time that the address is preferred; that is, i

unrestricted. When the preferred lifetime expires, the a becomes deprecated and its use is restricted.

### **reconfigure enumint(allow=1, disallow=2, require=3) default = allow**

Controls DHCPv6 client reconfiguration support:

- |   |          |                                                                                                                                                     |
|---|----------|-----------------------------------------------------------------------------------------------------------------------------------------------------|
| 1 | allow    | Allows clients to request reconfigura support and the server will honor the request (default).                                                      |
| 2 | disallow | Allows clients to request reconfigura support but the server will not honor the clients' request.                                                   |
| 3 | require  | Requires clients to request reconfigu support and the server drops client Solicit and Request messages that do include a Reconfigure-Accept option. |

This attribute has special handling during the policy hi processing when checking the Prefix policies (embedded o for the Prefixes on a Link. The Prefixes for the Link ar processed in alphabetic (case blind) order. For each Pre embedded and then named policy are checked. Only Prefixe the client has access (based on selection tags, etc.) ar as follows:

- If any of the prefix policies has this attribute se disallow or require, that setting is used.
- Otherwise, if at least one has it set to allow, Rec is allowed.
- If no prefix policies have this attribute set, the policies in the hierarchy are checked.

### **reconfigure-via-relay bool default = false**

Controls whether the server should prefer unicasting or relaying DHCPv6 Reconfigure messages.

If false (the default), the server prefers to unicast Reconfigure messages if the client has one or more valid statefully assigned addresses.

If true, the server prefers to send Reconfigure messages via the relay agent unless no relay agent information is available.

Note: When you use this attribute, consider that:

- In networks where the DCHPv6 server cannot commun directly with its client devices?for example, whe firewalls are in place?set this value to true.
- The DHCPv6 server does not use embedded and named policies configured on a client when it evaluates this attribute.
- The relay agent cannot be used if the Relay-Forw came from a link-local address.

### **reverse-dnsupdate nameref(DnsUpdateConfig)**

Specifies the name of the update configuration that dete which reverse zones to include in a DNS update.

### **server-lease-time time**

Tells the server how long a lease is valid. For more fr communication with a client, you might have the server c leases as leased for a longer period than the client con This also provides more lease-time stability. This value unless it is longer than the lease time in the dhcp-leas found through the normal traversal of policies.

### **split-lease-times bool default = disabled**

Specifies a value that the DHCP server might use interna

affect lease times.

If enabled, the DHCP server still offers clients lease times that reflect the configured lease-time option from the appropriate policy; but the server bases its decisions regarding expiration on the 'server-lease-time' value.

#### **unavailable-timeout** **time** default = 24h

Permits the server to make a lease unavailable for the time specified and then to return the lease to available state. If otherwise configured in the system\_default\_policy, then the default is 86400 seconds (or 24 hours).

#### **use-client-id-for-reservations** **bool** default = off

Controls how the server database checks for reserved IP addresses.

By default, the server uses the MAC address of the DHCP client key for its database lookup. If this attribute is set to on (enabled), then the server does the check for reserved addresses using the DHCP client-id, which the client supplies. In cases where the DHCP client does not supply the client-id, the server synthesizes it, and uses that value.

#### **v4-bootp-reply-options** **optionid4**

Lists the options the server returns to all BOOTP clients.

#### **v4-reply-options** **optionid4**

Lists the options the server returns to all DHCPv4 clients or not the client specifically asks for the option data.

#### **v6-reply-options** **optionid6**

Lists the options that should be returned in any replies to DHCPv6 clients.

This attribute has special handling during the policy hierarchy processing when checking the Prefix policies (embedded on the Prefixes on a Link). The Prefixes for the Link are processed in alphabetic (case blind) order. For each Prefix embedded and then named policy are checked. Only Prefixes the client has access (based on selection tags, etc.) are

#### **valid-lifetime** **time** default = 2w

Assigns the default and maximum valid lifetime for lease DHCPv6 client interfaces. Expressed in seconds and relative to the time the server sent the packet, this attribute specifies the length of time that an address remains valid. When the period of time expires, the address becomes invalid and unusable. The valid lifetime must be greater than or equal to the preferred lifetime.

## dhcp-dns-update

dhcp-dns-update - Configures a DNS Update object for DHCP

### Synopsis

```
dhcp-dns-update <name> create [<attribute>=<value>]
dhcp-dns-update <name> delete
dhcp-dns-update list
dhcp-dns-update listnames
dhcp-dns-update <name> show
dhcp-dns-update <name> set <attribute>=<value> [<attribute>=<value> ...]
dhcp-dns-update <name> get <attribute>
dhcp-dns-update <name> unset <attribute>

dhcp-dns-update <name> enable <attribute>
```

**dhcp-dns-update** <name> **disable** <attribute>

## Description

The `dhcp-dns-update` command configures dynamic DNS update configurations for DHCP. DHCP policies refer to this information to control the DNS update behavior for each client.

## Examples

## Status

## See Also

[policy](#)

## Attributes

### **backup-server-addr** [ipaddr](#)

Specifies the backup DNS server address that receives DN if the server specified in `server-addr` is down.

### **backup-server-key** [nameref](#)(Key)

Specifies the TSIG key used to process all dynamic DNS u `backup-server-addr`.

### **dns-host-bytes** [rangeint](#)(1-4)

Sets the number of bytes in a lease IP address to use wh `in-addr.arpa` names. The server forms names in the `in-add` by prepending these bytes of the address (in reverse ord reverse zone name. If unset, the server synthesizes an a value based on the scope?s subnet size.

### **dynamic-dns** [enumbyt](#) default = update-all

Controls whether the DHCP server should attempt to updat DNS server with the name and address information from le that are granted to requesting clients.

### **force-dns-updates** [bool](#) default = false

Determines whether the DHCP server retries a dynamic DNS whenever a client renews its lease, even if DHCP thinks update has already been completed successfully.

### **forward-zone-name** [dname](#)

Designates the DNS forward zone used to add a DHCP clien (A record).

### **host-name-generator** [expr](#)

Defines an expression that evaluates to the synthesized to be used. If undefined or the expression returns "", t configured host name synthesis is used. If the expressio a null value, no host name will be used.

### **max-dns-ttl** [time](#)

Indicates the maximum number of seconds the DHCP server records it acquired through dynamic updates. Default is This value sets a ceiling (or time to live) on how long DNS updates.

When the DHCP server adds a DNS record, it uses a TTL of third the lease time if it is between `min-dns-ttl` and `max-dns-ttl` values. If one third of the lease time is gr

than max-dns-ttl, the TTL value is set to max-dns-ttl.

**min-dns-ttl** [time](#)

Indicates the minimum number of seconds the DHCP server DNS records acquired through dynamic updates. This value shortest allowable time (or time to live) to keep DNS up. When the DHCP server adds a DNS record, it uses a TTL of third the lease time if it is between min-dns-ttl and max-dns-ttl values. If one third of the lease time is sm than min-dns-ttl, the TTL value will be set to min-dns-t

**name** [string](#) required,unique,immutable

Gives a unique name to the configuration object.

**reverse-zone-name** [dname](#)

Designates the DNS reverse (in-addr.arpa) zone that is u with PTR records.

If a reverse-zone-name is configured, DHCP always uses i Alternately, if DHCP server synthesize-reverse-zone feat enabled and if a reverse-zone-name is not configured, DH uses the lease IP address and the scope dns-host-bytes t automatically generate a reverse zone name.

**reverse-zone-prefix-length** [rangeint\(0-124\)](#)

Identifies the prefix length of the reverse zone for ip6 updates. The server forms the zone name using this value configured; otherwise the prefix length is determined fr Prefix. This value must be a multiple of 4 as ip6.arpa z on 4 bit (nibble) boundaries. If not a multiple of 4, it rounded up to the next higher multiple of 4.

**server-addr** [ipaddr](#)

Specifies the DNS server address that receives dynamic D

**server-key** [nameref\(Key\)](#)

Specifies the TSIG key used to process all dynamic DNS u server-addr.

**synthesize-name** [bool](#) default = disabled

Controls whether the DHCP server automatically creates D names for DHCP clients that do not provide names. The se can synthesize unique names for clients based on the 'sy name stem' attribute.

**synthetic-name-stem** [string](#) default = dhcp

Identifies the stem of the default host name to use if c supply host names.

**update-dns-first** [bool](#) default = disabled

Controls whether to update the DNS server before grantin

**update-dns-for-bootp** [bool](#) default = enabled

Controls whether the DNS server receives updates for BOO If the server is replying to a BOOTP request, and is off lease configured to perform DNS updates, it checks this attribute before beginning the DNS update. This att an administrator to prevent DNS updates for BOOTP client allowing updates for DHCP clients. If not configured, th setting is used (which defaults to true).

**use-dns-update-prereqs** [bool](#) default = true

If configured, the update ignores DHCP server level use-dns-update-prereqs. By default (enabled), the DHCP s adds prerequisites to DNS update messages to assure that

current client uses the domain name, before updating the with the current lease (IP Address).

If disabled, the DHCP server assumes that the current client is entitled to the domain name (and hence not add prerequisite DNS update message) and associates the client's current lease with that domain name.

Note: The DHCP server always adds prerequisites to the DNS update message while adding a new domain name record on a client acquiring a new lease and removing the domain name when client release its lease or the lease expires.

### **v6-host-name-generator** *expr*

Defines an expression that evaluates to the synthesized host name to be used. If undefined or the expression returns "", the configured host name synthesis is used. If the expression returns a null value, no host name will be used.

### **v6-synthetic-name-generator** *enumint*(*duid=1*, *hashed-duid=2*, *cablelabs-cablelabs-cm-mac-addr=4*) *default = hashed-duid*

Controls what the DHCP server appends to the synthetic name when synthesizing the fully-qualified domain name for a client *duid*

*duid*  
The raw client DUID. This is formatted as a hexadecimal string with a hyphen as separator between each octet.  
*hashed-duid*

The hashed client DUID. The rightmost 64-bits of the hash over the client's DUID appended with the domain name (in DNS wire format) is used to generate a 13 base 32 encoded string. This is the default method fallback method if the configured type is not possible (that is, the option needed to generate the select does not exist for the client).

*cablelabs-device-id*

The Cablelabs CL\_OPTION\_DEVICE\_ID option data is used, available. This is formatted as a hexadecimal string with a hyphen as separator between each octet. This might be used for names generated for DOCSIS 3.0 cable modems.

*cablelabs-cm-mac-addr*

The Cablelabs CL\_CM\_MAC\_ADDR option data is used, available. This is formatted as a hexadecimal string with a hyphen as separator between each octet. This might be used for names generated for CPEs behind a customer DOCSIS 3.0 cable modem (if the customer has multiple interfaces a name collision and disambiguation will likely require).

Note: Several of these methods may cause privacy concern if the DNS is accessible from the Internet.

## **dhcp-interface**

`dhcp-interface` - Configures the DHCP server's network interfaces

### **Synopsis**

```
dhcp-interface <name> create [<attribute>=<value>]
dhcp-interface <name> delete
dhcp-interface list
dhcp-interface listnames
dhcp-interface <name> show
dhcp-interface <name> set <attribute>=<value> [<attribute>=<value> ...]
dhcp-interface <name> get <attribute>
dhcp-interface <name> unset <attribute>
```

```
dhcp-interface <name> enable <attribute>
dhcp-interface <name> disable <attribute>
```

## Description

The dhcp-interface command configures network interfaces for use by the Network Registrar DHCP server. If there are no defined interfaces, the server discovers and uses all available interfaces on the system. When this list is present, the server uses only the available interfaces, if any, that match this list.

## Examples

## Status

## See Also

## Attributes

### address subnet

Specifies the IP address and subnet mask length of one or more interfaces that the DHCP server should use.

### ip6address prefix

Specifies the IPv6 address and prefix length of one or more interfaces that the DHCP server should use.

### multicast-addresses ip6addr default = ff02::1:2,ff05::1:3

Enables or disables the specified multicast addresses on the interfaces. The default multicast addresses are ff02::1:2 and ff05::1:3. DHCPv6 requires address ff02::1:2, if any interfaces are directly connected to the link associated with the address. The address ff05::1:3 is the default multicast address used by relay agents when relaying DHCPv6 requests.

### name string required,unique,immutable

Identifies the interface for DNS server use.

## dhcp-subnet

dhcp-subnet - Describes a contiguous range of IP address space which the DHCP server has allocated to a client.

## Synopsis

```
dhcp-subnet <dhcp-subnet-number> [show]
dhcp-subnet <subnet-number> get <attribute>

dhcp-subnet <subnet-number> force-available
```

## Description

The dhcp-subnet commands manipulate subnet objects that the DHCP server has leased to its clients. When the DHCP server receives a request for a subnet, it creates the subnet by subdividing its available address-blocks, and allocating the subnet to the client.

The dhcp-subnet commands apply only to subnet objects that have

been allocated to clients by the DHCP server.

## Examples

```
dhcp-subnet 10.10.1.0/24 show
```

## Status

## See Also

[dhcp-address-block](#)

## Attributes

### **address** [subnet](#) **immutable**

Displays the subnet address, including the mask.

### **all-vpns** [bool](#)

Provides failover configuration information for this subnet. If true, this subnet is configured to use the same failover configuration for all VPNs. Read only.

### **client-domain-name** [string](#)

Displays the domain name the client specified in its message (if any).

### **client-flags** [flags](#)(client-valid=1, client-id-created-from-mac-address=2)

Displays either client-valid or client-id-created-from-mac-address (the client ID was created for internal use from the client's MAC address).

### **client-host-name** [string](#)

Displays the host name that the client specified (if any).

### **client-id** [blob](#)

Displays the client-id of the subnet's client.

### **client-last-transaction-time** [date](#)

Displays the time when the client last contacted the DHCP server.

### **client-mac-addr** [macaddr](#)

Displays the MAC address which the client presented to the DHCP server.

### **expiration** [date](#)

Displays the expiration time of the subnet binding.

### **high-water** [int](#)

Displays the highest utilization level recorded since the last time that statistics were retrieved.

### **in-use-addresses** [int](#)

Displays the number of addresses currently being used by the DHCP server.

### **last-transaction-time** [date](#)

Displays the time at which the client last communicated with the DHCP server about this subnet. Read only.

### **relay-agent-option** [blob](#)

Displays the contents of the relay agent information option from the most recent client interaction. Read only.

### **selection-tags** [string](#)

Displays the selection-tag string that the client presented it last leased or renewed the subnet binding. Read only.

**state** **enumint**(none=0, available=1, other-available=2, offered=3, leased=4, € released=6, unavailable=7, pending-available=8)

Displays the state of the subnet.

**unusable-addresses** **int**

The number of addresses marked unusable.

**vpn-id** **int** default = 0

Identifies the VPN that contains this subnet.

## dns

dns - Configures and controls the DNS server

### Synopsis

```
dns disable <attribute>
```

```
dns enable <attribute>
```

```
dns get <attribute>
```

```
dns set <attribute>=<value> [attribute]=<value ...]
```

```
dns unset <attribute>
```

```
dns show
```

```
dns findRR -name <fqdn>|<addr>
```

```
dns findRR [-namePrefix <namePrefix>] [-rrTypes <rrTypeList>]  
           [-protected | -unprotected] [-zoneType forward | reverse  
           | primary | secondary | published | unpublished | ALL]
```

```
dns addRootHint <name> <addr> [<addr> ...]
```

```
dns removeRootHint <name>
```

```
dns listRootHints
```

```
dns addException <name> <addr> [<addr> ...]
```

```
dns removeException <name>
```

```
dns listExceptions
```

```
dns addForwarder <addr>
```

```
dns removeForwarder <addr>
```

```
dns listForwarders
```

```
dns flushCache
```

```
dns rebuildRR-Indexes
```

```
dns forceXfer secondary
```

```
dns scavenge
```

```
dns serverLogs show
```

```
dns serverLogs nlogs=<nlogs> logsize=<logsize>
```

```
dns getStats [performance | query | errors | security | maxcounters |  
             ha | ipv6 | all] [total | sample]
```

```
dns resetStats
```

```
dns getZoneCount [forward | reverse | primary | secondary | published |
```

```

        unpublished | ALL]
dns getRRCount [zone <name> | forward | reverse | primary | secondary |
        published | unpublished | ALL]

dns removecachedRR <name> [<type>] [<data>]

dns setPartnerDown

```

## Description

The `dns` command lets you configure the DNS server in the cluster.

```

dns findRR -name <fqdn>|<addr>
dns findRR [-namePrefix <namePrefix>] [-rrTypes <rrTypeList>]
        [-protected | -unprotected] [-zoneType forward | reverse
        | primary | secondary | published | unpublished | ALL]

```

Use the `findRR` commands to display the resource records for a specific domain name; or to display those matching a name prefix, a list of resource record types--whether protected or unprotected--and certain zone types.

```

dns addRootHint <name> <addr> [<addr> ...]
dns removeRootHint <name>
dns listRootHints

```

Use the `RootHint` commands to add or remove the names and addresses of the root servers. After you specify the root servers, Network Registrar queries them for their root name server records. These records are in turn used to resolve other names. As such, these values need not be exact, but must be accurate enough for the Network Registrar DNS server to retrieve the correct information.

The `addRootHint` command adds the name of a root server and the addresses (to which it is listening on) to the list that the DNS server uses to find the roots of the DNS name space.

The `removeRootHint` command removes a root server from the list.

The `listRootHints` command lists the root server information.

```

dns addException <name> <addr> [<addr> ...]
dns removeException <name>
dns listExceptions

```

Use the `exception` commands only if you do not want your DNS server to use the standard name resolution for querying root name servers for names outside the domain. The `exception` command lets you specify the resolution exception domains and the IP addresses of the associated servers.

The `addException` command adds a list of DNS servers to use to resolve names in a specified domain. This list allows you to override DNS the publicly configured names server for one or more domains.

The `removeException` command removes an entry for exceptional resolution of addresses within a domain.

The `listExceptions` command lists the domains that are configured

to have exceptional resolution of their names.

**dns addForwarder <addr>**

**dns removeForwarder <addr>**

**dns listForwarders**

Use the Forwarder commands to specify the addresses of any name servers that you want your Network Registrar DNS server to use as forwarders. Network Registrar forwards recursive queries to these servers before forwarding queries to the Internet at-large. You can use the exception command to override forwarding for specific domains.

The addForwarder command adds the address of a forwarder for this DNS server.

The removeForwarder command removes the address of a forwarder.

The listForwarders command lists the forwarders for this DNS server.

**dns flushCache**

The flushCache command flushes the persistently cached information in the DNS server. To remove cached information, to be removed reload the DNS server.

**dns rebuildRR-Indexes**

The rebuildRR-Indexes command rebuilds the resource record indexes.

**dns forceXfer secondary**

The forceXfer command forces full zone transfers for every zone whose type matches the type (primary or secondary) specified in the command, regardless of the SOA serial numbers, to synchronize DNS data store. If a normal zone transfer is already in progress, the forceXfer command schedules a full zone transfer for that zone immediately after the normal zone transfer finishes.

Note: The option for primary is not yet available.

**dns scavenge**

The scavenge command causes scavenging to occur on all primary zones that have scavenge enabled.

**dns serverLogs show**

**dns serverLogs nlogs=<nlogs> logsize=<logsize>**

The serverLogs show command displays the number of log files and the maximum size for each file.

The serverLogs command allows setting the two server logging parameters, nlogs and logsize. Either or both may be specified in the command, and changes will only occur to the one(s) specified. When setting logsize, a suffix of K or M indicates units of thousands or millions.

```
dns serverLogs nlogs=6 logsize=500K
dns serverLogs logsize=5M
```

Note: For these changes to take effect you must save the changes

and restart the server Agent.

```
dns getStats [performance | query | errors | security | maxcounters |  
             ha | ipv6 | all] [total | sample]
```

#### **dns resetStats**

The getStats command displays the requested DNS server statistics, either since the last reload or for the last sample period.

The resetStats commands returns the DNS activity counters (statistics) to zero.

```
dns getZoneCount [forward | reverse | primary | secondary | published |  
                 unpublished | ALL]
```

```
dns getRRCount [zone <name> | forward | reverse | primary | secondary |  
                published | unpublished | ALL]
```

The getZoneCount and getRRCount commands display the number of zones or resource records for the requested zones. By default, all published zones are reported.

```
dns removecachedRR <name> [<type>] [<data>]
```

The removecachedRR command allows the removal of non-authoritative RRs from both the (non-authoritative) persistent cache and the in-memory cache. An RR name-set can be removed by supplying only 'name'. An RR-set can be completely removed by specifying both 'name' and 'type'; a specific RR can be purged by supplying 'name', 'type' and 'data'.

```
dns setPartnerDown
```

The setPartnerDown command notifies the DNS server that its High Availability DNS partner server is down.

## Examples

## Status

## See Also

[server](#)

## Attributes

**activity-counter-interval** [rangetime](#)(10s-24h) default = 5m

Sets the period of time that server activity counters use to collect metrics.

Use this attribute together with the 'collect-sample-counters' attribute. Make sure the 'collect-sample-counters' is set to true (enable) to start sampling.

**activity-counter-log-settings** [flags](#)(total=1, sample=2, performance=3, query=4, security=6, maxcounters=7, ha=8, ipv6=9, server=10) default = total,performance,query,errors,security,maxcounters,ha,ipv6

Controls what activity counters a DNS server uses for logging. The possible flags are:

total	log the accumulated counters since reset start.
-------	-------------------------------------------------

sample	log counters for each sampling interval.
performance	log performance-related counters.
query	log query-related counters.
errors	log error-related counters.
security	log security-related counters.
maxcounters	log maxcounters-related counters.
ha	log HA-related counters.
ipv6	log IPv6-related counters.

**activity-summary-interval** [rangetime\(60s-24h\)](#) default = 5m

Sets the seconds between DNS activity summary log messages if the activity-summary attribute is enabled in the server log-settings. The default value is 5 minutes.

**auth-db-cache-kbytes** [int](#) default = 10240

Sets the kilobytes of internal (BTREE) cache that the auth database uses.

Note: This value is rounded up to the nearest 4KB boundary. For example, an 81KB value is rounded up to 84KB.

**axfr-multirec-default** [bool](#) default = on

Determines the default multi-record AXFR choice for foreign (remote) servers not found in remote server list. Default is true (enable). Only old version Bind 8.x and Microsoft NT version require this attribute to set false.

**cache-db-cache-kbytes** [int](#) default = 10240

Sets the kilobytes of internal (BTREE) cache that the cache uses.

Note: This value is rounded up to the nearest 4KB boundary. For example, an 81KB value is rounded up to 84KB.

**checkpoint-interval** [rangetime\(60m-1w\)](#) default = 3h

Sets the seconds that elapse between snapshots of zone information. DNS records this information in the Zone Checkpoint data.

**collect-sample-counters** [bool](#) default = true

Switches counter sampling on and off.

**default-negcache-ttl** [rangetime\(0-24h\)](#) default = 0

Sets the amount of time in seconds that the DNS server caches negative answers if there is no SOA record in the authority section of the reply.

The presence of an SOA record in the authority section of the reply overrides this attribute value.

For more details, see IETF RFC2308.

**delegation-only-domains** [dname](#)

Instructs the DNS server to expect a specified zone to return only delegations to authoritative nameservers when queried.

Other than records at the domain name itself, the specified zone must contain only NS records for each authoritative nameserver to which the subzone is delegated and the zone apex SOA record. For example, 'com.' is a domain that contains only delegations.

Use this attribute to filter out wildcard or synthesized records from authoritative nameservers whose zone is not of interest.

The server enforces the delegation-only nature of the zone list when it examines responses that are not from a forwarder or resolution exception server. The server corrects non-conforming answers to no-such-name responses. This

be enforced when the answer comes from a forwarder.

**exception-forwarding** **enumint**(forward-last=0, forward-first=1, forward-always=forward-last)

Controls the order (forward-first, forward-last or forward-always) with which the DNS server will forward a query to a configured resolution exception.

forward-always - DNS will always forward queries to a configured resolution exception

forward-first - DNS will first forward queries to a configured resolution exception and if it doesn't get an answer before the request expires, it will forward to cached name servers (if any)

forward-last - DNS will first forward queries to cached name servers (if any) and if it doesn't get an answer before the request expires, it will forward to a configured resolution exception

**fake-ip-name-response** **bool** default = enabled

Controls whether the DNS server rejects queries of fully qualified domain names that are in IP address form (for example, an A record like 192.168.40.40), without even trying to query (or forward to) other servers. Default is true (enable).

Enabling this feature causes the server to respond to those queries indicating the name does not exist. Queries of this type are generally from rogue applications.

**forward-retry-time** **rangetime**(1s-30s) default = 8s

Sets the retry interval for forwarding a DNS query to a forwarder or resolution exception server. These queries are recursive, and may require more time for the forwarder to resolve.

To ensure the server tries all forwarders, set this value to 'request-expiration-time' divided by one less than the number of configured forwarders).

Note: This attribute has no effect when you enable slave; the server uses slave-forward-retry-time instead.

**ha-dns-comm-timeout** **rangetime**(2s-24h) default = 30s

Sets the period of time that must elapse before the server determines it cannot communicate with its HA DNS partner.

**ixfr-enable** **bool** default = enabled

Controls the incremental transfer behavior for zones for which you have not configured a specific behavior. Requires true (enable).

**ixfr-expire-interval** **rangetime**(0-68y5w3h14m7s) default = 0

Specifies the maximum duration between full zone transfers. After the DNS server will request incremental transfer updates for secondary zones. After this time interval expires, even if incremental transfer would normally be requested, the DNS server will instead request a full zone transfer. Setting this to 0 does not enforce any time limit between full zone transfers.

**local-port-num** **rangeint**(1-65535) default = 53

Specifies the UDP and TCP port number that the DNS server uses to listen for queries.

**log-settings** **flags**(config=1, ddns=2, xfr-in=3, xfr-out=4, notify=5, datastore=6, scavenger=9, scavenger-details=10, server-operations=11, lame-delegation=12, query=14, ddns-refreshes=15, ddns-refreshes-details=16, ddns-details=17, ts

details=19, activity-summary=20, query-errors=21, config-details=22, incoming-packets=23, outgoing-packets=24, xfr-in-packets=25, query-packets=26, notify-packets=27, ddns-packets=28, xfr-out-packets=29, lame-details=30) default = config-in, xfr-out, notify, datastore, scavenge, server-operations, root-query, tsig, query-errors-details

Determines which detailed events the DNS server logs, as using a bit mask. Logging these additional details can help analyze a problem. Leaving detailed logging enable for a period, however, can fill the log files and cause the loss of important information.

The possible flags are:

config

This flag will cause log messages pertaining to server configuration and server de-initialization (unconfig)

ddns

This flag will cause the logging of high-level DDNS. Detailed DDNS logging (e.g. RRs that have been deleted) can be enabled via ddns-details.

xfr-in

This flag will allow the generation of log messages associated with inbound full and incremental zone transfers.

xfr-out

This flag will allow the generation of log messages associated with outbound full and incremental zone transfers.

notify

This flag allows log messages associated with the processing of notify messages.

datastore

This flag allows the generation of log messages associated with datastores processing. Enabling this flag provides insight into various events in the server's embedded databases/datastores.

scavenge

This flag allows the logging of events associated with scavenging.

scavenge-details

This flag causes more detailed logging, pertaining to scavenging, to be displayed. Note, this flag is disabled by default.

server-operations

This flag enables the logging of general high server activity (such as those pertaining to sockets and interfaces)

lame-delegation

This flag allows the logging of lame-delegation encounters. Though enabled by default, disabling this flag could prevent the log from getting filled with frequent lame-delegation encounters.

root-query

This flag allows the generation of log messages associated with queries (and responses) from root servers.

ddns-refreshes

This flag allows the server to log messages associated with (W2K) DDNS refreshes.

ddns-refreshes-details

This flag generates log messages that provide detail describing RRs that were refreshed.

ddns-details

This flag enables detailed logging that describes the

that have deleted/added due to DDNS updates.

**tsig**  
This flag allows the logging of events associated Tr Signature (TSIG) DDNS updates.

**tsig-details**  
This flag causes more detailed logging, pertaining t be displayed. Note, this flag is disabled by default

**query-errors**  
This flag causes logging of errors encountered while processing DNS queries.

**config-details**  
This flag generates detailed information during serv configuration (e.g. displaying all configured and as server attributes)

**incoming-packets**  
This flag causes incoming packets to be traced.

**outgoing-packets**  
This flag causes outgoing packets to be traced.

**xfr-in-packets**  
This flag causes incoming zone transfer packets to b

**xfr-out-packets**  
This flag causes outgoing zone transfer packets to b

**query-packets**  
This flag causes query packets to be traced.

**notify-packets**  
This flag causes notify packets to be traced.

**ddns-packets**  
This flag causes DDNS packets to be traced.

**performance**  
This flag logs server level performance statistics.

**ha-details**  
This flag enables detailed logging of HA related inf

**max-cache-ttl** [rangetime](#)(0-68y5w3h14m7s) default = 1w

Sets the maximum amount of time that you want the DNS se to retain cached information. Default is 604800s (1w).

**max-dns-packets** [int](#) default = 500

Specifies the maximum number of packets that dns server handle concurrently. DNS server will drop inbound packet limit is reached.

**max-negcache-ttl** [rangetime](#)(0-68y5w3h14m7s) default = 60m

Sets an upper limit on the number of seconds a DNS serve maintains negative cache entries. Default is 3600s (1h). A server generates these cache entries after querying ar name server and receiving an authoritative reply stating requested records do not exist.

**mem-cache-size** [rangeint](#)(200-4194303) default = 50000

Indicates the size, in kilobytes, of the in-memory recor Default is 50000 (50MB).

**minimal-responses** [bool](#) default = false

Controls whether the DNS server omits or includes record the authority and data sections of query responses when records are not required. Enabling this attribute may in query performance such as when the DNS server is configu a caching server.

**neg-cache-ttl** [rangetime](#)(0-68y5w3h14m7s) default = 10m

Indicates how long information learned from other name

servers about non-existent names or data should be cache

**no-fetch-glue** **bool** default = disabled

Specifies whether you want the DNS server, when composing a response to a query, to fetch missing glue records. Glue records are DNS A records, which specify the address domain's authoritative name servers. Normal DNS response NS records and their A records related to the name being

**no-recurse** **bool** default = disabled

Specifies whether you want to disable forwarding client to other name servers when your DNS server is not authoritative for data in its own cache. If you disable recursive queries, you make your name server a noncaching

**notify** **bool** default = enabled

Controls how the DNS server sends NOTIFY packets for zones that have changed. Default is true (enable). You must also set these attributes or accept their default values: notify-defer-cnt, notify-min-interval, notify-rcv-interval, notify-send-stagger, notify-wait.

**notify-defer-cnt** **int** default = 100

With the notify attribute enabled, sets the maximum number of changes the DNS server can accumulate during the notify-period. If changes exceed this number, the DNS server sends notification before the notify-wait period has elapsed. Default is 100 changes.

**notify-min-interval** **rangetime**(0-68y5w3h14m7s) default = 2s

With the notify attribute enabled, sets the minimum interval required before sending notification to a particular server for consecutive changes on the same zone. Default is 2s.

**notify-rcv-interval** **rangetime**(0-68y5w3h14m7s) default = 5s

With the notify attribute enabled for secondary zones, sets the minimum amount of time between complete processing of one notification (serial number testing and/or zone transfer) and the start of processing of another notification. Default is 5s.

**notify-send-stagger** **rangetime**(0-68y5w3h14m7s) default = 1s

With the notify attribute enabled, sets the interval to use for staggering notifications to multiple servers about a zone change. Default is 1s.

**notify-source-address** **ipaddr**

Specifies the source IP address that the DNS server uses for notify requests to other servers. A value of 0.0.0.0 indicates that the operating system uses the best local address based on the destination.

**notify-source-port** **rangeint**(0-65535)

Specifies the UDP port number that the DNS server uses for notify requests to other servers. A value of zero indicates that the server should choose the port. If this attribute is unset, then queries are sent to the port used to listen for queries (See the local-port-attribute).

**notify-wait** **rangetime**(0-68y5w3h14m7s) default = 5s

With the notify attribute enabled, and after an initial change, sets the period of time for the DNS server to wait before it sends change notification to other name servers

Default is 5s.

This property allows you to accumulate multiple changes, and thus limit the number of times the serial number adv

**persist-mem-cache** `bool` default = true

Specifies whether the DNS server writes memory cache, c it from the persistent cache database.

If you set this to false, DNS does not use the persister database.

**query-source-address** `ipaddr`

Specifies the source IP address from which the DNS server send queries to other servers when resolving names for c value of 0.0.0.0 indicates that OS will use the best loc based on the destination.

**query-source-port** `rangeint(0-65535)`

Specifies the UDP port number that the DNS server uses t queries to other servers when resolving names for client value of zero instructs the server to choose a random pc you unset this attribute, the server sends queries from port it uses to listen for queries (see the local-port-r attribute).

**remote-port-num** `rangeint(1-65535)` default = 53

Specifies the UDP and TCP port number the DNS server uses to send queries to other servers. Default is port 5

**request-expiration-time** `rangetime(10s-5m)` default = 1m30s

Governs the expiration time of DNS queries; for example, DNS query, zone transfer SOA query, IXFR request and not request). A query that is not answered within this time expires.

Note: Cisco recommends that you make sure this value is considerably larger than 'request-retry-time' to allow n attempts to query multiple servers, using exponential ba

**request-retry-time** `rangetime(1s-30s)` default = 4s

Dictates the retry time interval (in secs) when querying server. This time interval is used in general queries - response to DNS clients queries - (zone transfers) SOA c IXFR requests and notify requests. Understand that this minium retry time. The server applies an exponential bac retries.

**restrict-cache-query-acl** `amelist`

Specifies an access control list that restricts which clients are allowed to query for cached non-authoritativ resource records. If unset, restrict-query-acl is used. Optional.

**restrict-query-acl** `amelist` default = any

Provides a global access control list (ACL) used to limi queries that a DNS server must honor. You can restrict c clients based on host IP address, network address, TSIG and access control lists. The default is to allow any cl perform a query.

Zones inherit this ACL if they are missing their restrict-query-acl. This ACL also serves as filtering qu non-authoritative zones.

**restrict-recursion-acl** `amelist` default = any

Defines the global Access Control List (ACL) used to res recursive that the DNS server honors. This list can cor

host, network addresses, TSIG keys and global ACLs that restrict recursive queries to a certain set of DNS client. Default is to allow any client to perform a query. For a not passing this ACL and requesting a recursive query, the server responds with a referral - as if originally requested with an iterative query.

**restrict-xfer-acl** [amelist](#) default = none

Overrides the default access control list (designating which clients can receive zone transfers).

**round-robin** [bool](#) default = enabled

Specifies whether you want round-robin cycling of equivalent records in responses to queries. Equivalent records are of the same name and type. Since clients often only look at the first record of a set, enabling this feature can help balance loads and keep clients from forever trying to talk to an out-of-service host.

**save-negative-cache-entries** [bool](#) default = enabled

Controls whether to have the server store negative query responses. Default, true (enable).

Disabling this feature prevents persistent caching of negative query responses.

**scvlg-ignore-restart-interval** [rangetime](#)(2h-24h) default = 2h

Ensures that the server does not reset the scavenging timer at every server restart. Within this interval, Network Registrar ignores the time between when a server went down and its restart. This interval is normally short. The value can range from a few hours to one day. With any time longer than that set, Network Registrar recalculates the scavenging period to allow for record updates that cannot take place while the server is stopped. You can also set this attribute on a server and the value set on the zone overrides the server setting. Default is 2h.

**scvlg-interval** [rangetime](#)(60m-1y) default = 1w

Sets the number of seconds that DNS waits before removing (scavenging) out-of-date address (A) records.

**scvlg-no-refresh-interval** [rangetime](#)(60m-1y) default = 1w

Sets the number of seconds during which DNS updates cannot be made to the zone timestamp.

**scvlg-refresh-interval** [rangetime](#)(60m-1y) default = 1w

Sets the number of seconds during which DNS updates cannot be made to the zone timestamp. After both the no-refresh and refresh intervals expire, the record is a candidate for scavenging. The value can range from one hour to 365 days. The zone setting overrides the server setting of 604800s (1w).

**simulate-zone-top-dynupdate** [bool](#) default = disabled

Enables compatibility with a Windows 2000 Domain Controller. When processing a dynamic update packet, which attempts to add or remove A records from the name of a zone, DNS responds that the update succeeded, rather than responding with a refusal would normally occur due to the protected/unprotected name conflict. No update to the records at the zone name will occur, although the response indicates that it has.

**slave-forward-retry-time** [rangetime](#)(5s-60s) default = 30s

Sets the retry interval for forwarding a DNS query in slave mode. These queries are recursive, and may require more time for

forwarder to resolve.

To ensure the server tries all forwarders, set this value 'request-expiration-time' divided by one less than the number of configured forwarders.

### **slave-mode** `bool` default = disabled

Specifies whether you want this server to be a slave server, a server that relies entirely on forwarders for data that is not in its cache. Default is false (disabled).

This attribute takes effect only if you specify the corresponding forwarders.

Note: You can override slave-mode for specific domains with the exception-list property.

### **subnet-sorting** `bool` default = disabled

Controls whether DNS reorders A records when responding to queries. Default is false (disabled).

As implemented in BIND 4.9.7, the Network Registrar DNS server confirms the client's network address before responding to a query. If the client, server, and target of the query are on the same subnet, and the target has multiple A records, the server reorders the A records in its response by putting the closest address first in the response packet.

Because clients often only look at the first record in a response, enabling this attribute can help localize network traffic to a subnet. This attribute is only applied on answers to queries from clients located on the same subnet as the DNS server.

### **tcp-query-retry-time** `rangetime(10s-5m)` default = 10s

Defines the retry time for DNS queries over a TCP connection. Cisco recommends that you set this value to less than the 'request-expiration-time'.

### **transfer-source-address** `ipaddr`

Specifies the source IP address that the DNS server uses for transfer and SOA requests to other servers.

A value of 0.0.0.0 indicates that the operating system uses the local address based on the destination.

### **transfer-source-port** `rangeint(0-65535)`

Specifies the UDP port number that the DNS server uses for transfer and SOA requests to other servers.

A value of 0 (zero) indicates that you should choose a random port. If this attribute is unset, then queries are sent from the port used to listen for queries (see local-port-num).

### **traps-enabled** `flags`(all=1, server-start=2, server-stop=3, ha-dns-partner-down=4, partner-up=5, ha-dns-config-error=6, masters-not-responding=7, masters-resp=8, secondary-zone-expired=9, forwarders-not-responding=10, forwarders-resp=11)

Defines the traps that this server is configured to send.

- 1 all  
Sends notifications for all server events.
- 2 server-start  
Sends notifications whenever the server is started or reinitialized.
- 3 server-stop  
Sends notifications whenever the server is stopped.
- 4 ha-dns-partner-down  
Sends notifications whenever the HA DNS partner goes down.
- 5 ha-dns-partner-up  
Sends notifications whenever the HA DNS partner becomes available.

- available again after going down.
- 6 `ha-dns-config-error`  
Sends notifications when a configuration mismatch HA DNS partners occurs.
- 7 `masters-not-responding`  
Sends notifications when master servers stop resp
- 8 `masters-responding`  
Sends notifications when master servers start resp again.
- 9 `secondary-zone-expired`  
Sends notifications when a secondary server can no claim authority for zone data when responding to c during a zone transfer.
- 10 `forwarders-not-responding`  
Sends notifications when DNS forwarders stop resp
- 11 `forwarders-responding`  
Sends notifications when DNS forwarders start resp again.

### **update-acl** `amelist` default = none

Provides server-level control of which devices can access update the DNS server. If the access control list is set zone level, Network Registrar overrides the server-level

### **update-relax-zone-name** `bool` default = disabled

Enables DNS updates to specify any zone name in the auth zone rather than the exact zone name; thus, relaxing the restriction on the zone name record in dynamic updates. attribute allows updates to specify a zone name which is within an authoritative zone rather than exactly the name

## dns-interface

`dns-interface` - Configures the DNS server's network interfaces

### Synopsis

```

dns-interface <name> create [<attribute>=<value>]
dns-interface <name> delete
dns-interface list
dns-interface listnames
dns-interface <name> show
dns-interface <name> set <attribute>=<value> [<attribute>=<value> ...]
dns-interface <name> get <attribute>
dns-interface <name> unset

dns-interface <name> enable <attribute>
dns-interface <name> disable <attribute>

```

### Description

The `dns-interface` command configures network interfaces for use by the Network Registrar DNS server. If there are no defined interfaces, the server discovers and uses all available interfaces on the system. When this list is present, the server uses only the available interfaces, if any, that match this list.

### Examples

### Status

## See Also

## Attributes

### **address** *subnet*

Specifies the IP address and subnet mask of the DNS interface.

### **ip6address** *prefix*

Specifies the IPv6 address and prefix length for one of the DNS interfaces.

### **name** *string* **required,unique,immutable**

The name of the DNS server interface.

### **port** *rangeint(1-65535)* **default = 53**

Specifies the UDP and TCP port number the DNS server listens on.

## dns-update-map

**dns-update-map** - Configures a DNS update map of the DHCP and DNS server configurations needed to perform DNS updates

## Synopsis

```
dns-update-map <name> create <dhcp-servers> <dns-servers>
                                     <dns-config> [<attribute>=<value> ...]

dns-update-map <name> delete

dns-update-map list

dns-update-map listnames

dns-update-map <name> show

dns-update-map <name> get <attribute>

dns-update-map <name> set <attribute>=<value> [<attribute>=<value> ...]

dns-update-map <name> unset <attribute>

dns-update-map <name> push
```

## Description

The **dns-update-map** command lets you define and manage DNS update configuration maps. A DNS update map defines an update relationship between a DHCP policy and a list of DNS zones. The update map is designed to coordinate:

- DNS servers or Highly Available (HA) DNS server pairs
- DNS update ACLs or update policies
- DHCP servers or failover server pairs
- DHCP policy selection

An update map applies to all primary zones that the DNS server manages, and all scopes that the DHCP server manages.

## Examples

## Status

## See Also

[dhcp](#), [failover-pair](#), [dns](#), [ha-dns-pair](#), [dhcp-dns-update](#), [acl](#), [update-po](#)

## Attributes

### **dhcp-client-class** [nameref](#)(ClientClass)

If `dhcp-policy-selector=use-client-class-policy` specifies the named `client-class` embedded policy should specify the `DnsUpdateConfig` referenced. A new embedded policy will be created if one does not exist when the map is applied.

### **dhcp-named-policy** [nameref](#)(Policy)

If `dhcp-policy-selector=use-named-policy`, this specifies the named policy which should specify the `DnsUpdateConfig` referenced in this map.

### **dhcp-policy-selector** [enumint](#)(`use-named-policy=1`, `use-client-class-policy=3`) default = 1, required

Indicates how to find the DHCP Policy to which `DnsUpdateConfig` referenced in the `dns-config` a `named-policy` or `client-class-embedded-policy` in the referenced policy and/or `client-class` must be on the DHCP servers before the map is applied. If `scope-embedded-policy` is selected, it is applied on the DHCP servers. A new embedded policy is created if it does not exist when the map is applied.

### **dhcp-servers** [oid](#) required

The DHCP server or DHCP failover pair associated with this configuration.

### **dns-config** [nameref](#)(DnsUpdateConfig) required

The `DnsUpdateConfig` object associated with this configuration.

### **dns-servers** [oid](#) required

The DNS server or HA pair associated with this configuration.

### **dns-update-acl** [amelist](#)

The update-acl to apply to zones referenced by `DnsUpdateConfig` in this map. If this attribute is set, then this attribute is ignored. If neither attribute is set, then a simple update-acl is constructed enabling just the `dhcp-servers` to update using the IP address(es) of the single DHCP server and, if specified, the server-key from the `DnsUpdateConfig` in the `dns-config` attribute.

### **dns-update-policy-list** [nameref](#)(UpdatePolicy)

The list of DNS update policies to apply to zones referenced by `DnsUpdateConfig` in this map. If the attribute is set, then this attribute is ignored. If neither attribute is set, then a simple update-acl is constructed enabling just the `dhcp-servers` to update using the IP address(es) of the single DHCP server and, if specified, the server-key from the `DnsUpdateConfig` in the `dns-config` attribute.

### **name** [string](#) required,unique

The name of this managed DNS update configuration.

# exit

`exit` - Exits the current nrcmd session.

## Synopsis

`exit`

## Description

The `exit` command lets you exit the current nrcmd session. If you have unsaved changes, they will be flushed to the database before the session exits.

## Examples

## Status

## See Also

[save\(nrcmd\)](#)

# export

`export` - exports configuration information to a file

## Synopsis

```
export leases -server
  [ -vpn <vpn-name> ]
  [ -time-ascii | -time-numeric ] <file>

export leases -client
  [ -vpn <vpn-name> ]
  [ -time-ascii | -time-numeric ] <file>

export zone <zone name> {ccm | dns | both} [<file>]

export zonenames {forward | reverse | both} [<file>]

export hostfile [<file>]

export addresses [file=<CSV text file> | { database=<data source name>
  user=<db user name> password=<db user password>
  [table=<table name>] } ] [config=<config file>]
  [vpn=<vpn-name>]
  [dhcp-only] [time-ascii | time-numeric]

export keys <file>

export key <keyname> <filename>

export option-set <option-set name> <filename>
```

## Description

The `export` command lets you export data in one of several common formats.

```
export leases -server
```

[ **-vpn** <vpn-name> ]  
[ **-time-ascii** | **-time-numeric** ] <file>  
Use the export leases **-server** command to export the current and expired leases to a file in the DHCP server's log directory.

**export leases -client**

[ **-vpn** <vpn-name> ]  
[ **-time-ascii** | **-time-numeric** ] <file>  
Use the export leases **-client** command to export the current leases to the specified file or the standard output.

**export zone** <zone name> {**ccm** | **dns** | **both**} [<file>]

Use the export zone command to export the resource records in the specified zone. BIND can parse the format of the output.

**export zonenames** {**forward** | **reverse** | **both**} [<file>]

Use the export zonenames command to export the list of zones that match a given criteria.

**export hostfile** [<file>]

Use the export hostfile command to export the CCM DNS information in the cluster in a UNIX hostfile format.

**export addresses** [**file**=<CSV text file> | { **database**=<data source name> **user**=<db user name> **password**=<db user password> [**table**=<table name>] } ] [**config**=<config file>]

[**vpn**=<vpn-name>]  
[**dhcp-only**] [**time-ascii** | **time-numeric**]  
Use the export addresses command to export DNS and IP Address information to a CSV (comma-separated value) text file, or the standard output, or to an ODBC database.

Output of the export command can include the VPN specification. The value can be any valid, predefined VPN name, or the reserved words global and all.

global--indicates all addresses not in any of the defined VPNs.  
all--indicates all VPNs, including the global one.

If you omit the VPN, the current one applies, as set by the session set current-vpn command. If the current VPN is undefined, the global VPN applies. Network Registrar adds the ID of the VPN at the end of each output line in the export file.

Use these conventions for export addresses keywords:

Configuration file

If it exists, the default configuration file is .nrconfig. To use a configuration file other than the default file, use the config keyword to identify your configuration file. If there is an [export-addresses] section in the configuration file, the export command uses the clusters that the section specifies instead of the default cluster. If you omit a configuration file, the export addresses command looks for a default .nrconfig file. This is the same configuration file that the report command uses.

Network Registrar looks for the file first in your current directory,

then in your home directory, and finally in the install-path/conf directory. It uses the first file it encounters. Each line of the configuration file must begin with the character # (comment), a section header enclosed in square brackets, or a parameter=value pair or its continuation. For example:

```
[export addresses]
  clusters=machine1 username password, machine2 username password [...]
```

Network Registrar strips leading white space from each line and ignores blank lines.

#### dhcp-only

This keyword causes the command to output only DHCP information and not DNS information.

#### Database tables

The table keyword specifies the database table to which the command exports address information. If you omit this keyword, Network Registrar writes to the default table name ip\_addresses. If the table already exists in the specified database when you run the export command, Network Registrar clears (and resets the columns) before writing the new data. Network Registrar does not provide a warning or confirmation if it clears an existing table.

#### Date and time

The optional time-ascii and time-numeric keywords specify how to output date/time fields to a CSV text file and when the target database does not support the timestamp data type. The default is time-ascii.

If no vpn is specified, the current-vpn of the session is used. is used.

When specifying a <vpn-name> to any export command which supports it, the name "global" (with or without the quotes) will specify the global (i.e., unnamed or default) vpn. The name "all" (also with or without the quotes) will specify that all vpn (including the global one) should be exported.

You can use the export command to export the TSIG keys that are configured on a cluster with the export keys command. You can also specify to export a single key with the export key <keyname> command. These commands will generate key definitions in BIND syntax so they may be either imported into other clusters or copied into BIND configurations.

The export option-set command writes out a text file that may be loaded into a running server with the option-set import command.

## Examples

## Status

## See Also

[session](#) current-vpn

# extension

extension - Integrates user-written DHCP extensions into the DHCP server

## Synopsis

```
extension list
extension listnames
extension <name> create <lang> <file> <entry> [<attribute>=<value>...]
extension <name> delete
extension <name> get <attribute>
extension <name> set <attribute>=<value> [<attribute>=<value> ...]
extension <name> unset <attribute>
extension <name> show
```

## Description

The extension command lets you configure extension modules. To extend the DHCP server with an extension module, you must do the following:

1. Write the extension module in either Tcl, C or C++, and install it in the server scripts directory.
2. Configure the DHCP server to recognize this extension, using this command.
3. Attach the configured extension to one or more DHCP script points using the dhcp attachExtension command.

## Examples

```
Create a tcl script, sample1, that does something.
nrcmd> extension sample1 create tcl sample1.tcl sample_ext
```

## Status

## See Also

[dhcp AttachExtension](#)

## Attributes

### entry [string](#) required

Identifies the entry point for the module. This function is called from any extension point to which this module is

### file [string](#) required

Provides the filename relative to the directory of the DHCP installation, or as an absolute pathname, but this cannot be a sequence of two dots (..).

### init-args [string](#)

Describes the arguments that should be passed to the init-entry point function. The arguments are passed in the environment dictionary using the key "Arguments".

### init-entry [string](#)

Specifies the name of the init entry point. If set,

calls this function when the server loads the module.

**lang** [enumstr](#)(Tcl=1, Dex=2)

Describes the language in which the script or module is. 'Tcl' indicates that the module is a Tcl script (tcl8. indicates that the module is a shared object with C ca interfaces.

**name** [string](#) required,unique,immutable

Designates the script or module. The DHCP server uses this name to assign scripts or modules to script point

## failover-pair

failover-pair - configures a DHCP failover relationship

### Synopsis

```
failover-pair <name> create <main-cluster/address>
                        <backup-cluster/address>
                        [<attribute>=<value> ...]
                        [addMatch [<vpn>/]<address/mask>]

failover-pair <name> delete
failover-pair list
failover-pair listnames
failover-pair <name> show

failover-pair <name> get <attribute>
failover-pair <name> set <attribute>=<value> [<attribute>=<value> ...]
failover-pair <name> unset <attribute>

failover-pair <name> addMatch [<vpn>/]<address/mask>
failover-pair <name> removeMatch [<vpn>/]<subnet/mask>
failover-pair <name> listMatches

failover-pair <name> sync <update | complete | exact>
                        [<main-to-backup | backup-to-main>]
```

### Description

The failover-pair command lets you define and manage the failover relationship between a main and backup server.

Either the main and backup clusters or the main and backup server IP addresses can be specified with the create command. If the main-server and backup-server addresses are set, the cluster addresses will only be used for synchronization of the server configuration. The referenced clusters must be configured with appropriate connection credentials for the sync command to be successful.

### Examples

### Status

### See Also

[cluster](#)

### Attributes

**backup oid**(CCMCluster)

Identifies the cluster that contains the backup server a failover pair.

**backup-pct percent** default = 10%

Controls the percentage of available addresses that the server sends to the backup server. Set this value on the main server. If it is set on a backup server, it is ignored (to enable copying of configurations).

Unless you explicitly set this value on a scope and you have load balancing, the value set here becomes the default

**backup-server ipaddr**

Controls the IP address used for the failover protocol backup server. If this value is unset, the address specified for the backup cluster is used. Cisco recommends setting this attribute only if the server is configured with different interfaces for configuration management and client requests. Always configure the failover protocol with the interface used to serve clients.

**dynamic-bootp-backup-pct percent**

Determines the percentage of available addresses that the main server sends to the backup server for scopes on which BOOTP is enabled. If defined, it must be defined on the main server. If it is defined in a backup server, it is ignored (to enable copying of configurations). If it is defined at all or the value is 0, the "backup-pct" is used instead. This parameter is separate from "backup-pct" because if dynamic BOOTP is enabled on a scope, a server will never, even in PARTNER-DOWN state, grant leases on addresses that are available to the other server because they can never safely be assumed to be available again. The MCLT has no meaning for dynamic BOOTP leases.

**failover bool** default = true

Enables failover configuration. If you disable this attribute you turn off failover on attached subnets without changing configuration fundamentals.

**load-balancing bool** default = disabled

Determines whether load balancing (RFC 3074) is enabled on a pair. The default is disabled. When enabled, the backup server is ignored and the main and backup server evenly split the load and available leases for all scopes in the failover relationship (that is, as if backup-pct were configured

**main oid**(CCMCluster)

Identifies the cluster with the main server for a failover

**main-server ipaddr**

Controls the IP address used for the failover protocol main server. If this value is unset, the address specified for the main cluster is used. Cisco recommends setting this attribute only if the server is configured with different interfaces for configuration management and client requests. Always configure the failover protocol with the interface used to serve clients.

**mclt time** default = 60m

Sets the maximum client lead time in seconds. This attribute controls how far ahead of the backup server that you can

the client lease expiration. You must define this value for the main and backup servers, and make sure the value is identical on both servers.

**name** [string](#) required,unique,immutable

Names a failover pair.

**persist-lease-data-on-partner-ack** [bool](#) default = true

Controls when the main server updates its database. Normally the main server updates its database when the backup server ACKs it with what the backup knows. Disabling this capability speeds up the main server, but after a restart the main server is out of sync with what the backup knows, and may offer clients one lease period with a renew time of the current lease plus the MCLT.

**poll-lease-hist-interval** [rangetime](#)(0-1y)

Specifies how often to collect lease history from the DHCP server for this cluster. If set to 0, no poll occurs.

**poll-lease-hist-offset** [rangetime](#)(0-24h)

Provides a fixed time of day for lease history polling. This time is interpreted as a time of day offset, with 12 midnight, provided the polling interval is less than the offset value is greater than the polling interval, or the polling interval is greater than 24 hours, the offset will be ignored. The scheduler for polling will ensure that the first poll event occurs at the offset time. For example, if you set the polling interval to 4 hours and the offset to 2am, the polling events occur at 2am, 6am, 10am, 2pm, 6pm and 10pm.

**poll-lease-hist-retry** [rangeint](#)(0-4)

Specifies how often to retry if the server fails to poll the data.

**poll-lease-hist-server-first** [enumint](#)(mainserver=0, backupserver=1) default

Specifies which server to poll first:

- 0 main server
- 1 backup server

**poll-subnet-util-interval** [rangetime](#)(0-1y)

Specifies how often to collect the subnet utilization data from the DHCP server for this cluster. If set to 0, no poll occurs.

**poll-subnet-util-offset** [rangetime](#)(0-24h)

Provides a fixed time of day for subnet utilization polling. This time is interpreted as a time of day offset, with 12 midnight, provided the polling interval is less than the offset value is greater than the polling interval, or the polling interval is greater than 24 hours, the offset will be ignored. The scheduler for polling will ensure that the first poll event occurs at the offset time. For example, if you set the polling interval to 4 hours and the offset to 2am, the polling events occur at 2am, 6am, 10am, 2pm, 6pm and 10pm.

**poll-subnet-util-retry** [rangeint](#)(0-4)

Specifies how often to retry if the server fails to poll the data.

**poll-subnet-util-server-first** [enumint](#)(mainserver=0, backupserver=1) default  
mainserver

Specifies which server to poll first:

```
0 main server
1 backup server
```

**safe-period** **time** default = 24h

Controls the safe period, in seconds. It does not have the same on both main and backup servers. It only has if use-safe-period is enabled. Define this attribute on main server. If it is defined on a backup server, it is ignored (to enable copying of configurations).

**scopetemplate** **oid**

Associates a scope template with a specified failover p

**use-safe-period** **bool** default = disabled

Controls whether a server can enter PARTNER-DOWN state an operator command. If disabled, a server never enters PARTNER-DOWN without an operator command. Define this attribute on the main server. If it is defined on a backup server, it is ignored (to enable copying of configurations).

## group

group - Configures a named group of administrators

### Synopsis

```
group <name> create [<attribute>=<value>]
group <name> delete
group list
group listnames
group <name> show
group <name> set =<value> [<attribute>=<value> ...]
group <name> get <attribute>
group <name> unset <attribute>

group <name> enable <attribute>
group <name> disable <attribute>
```

### Description

The group command configures the specified group of administrators. Administrator groups are used to associate an admin to one or more roles that control access to operations and data.

### Examples

### Status

### See Also

[admin](#), [role](#)

### Attributes

**desc** **string**

Describes this group.

**name** **string** required,unique

Names this group of administrator roles.

## ha-dns-pair

ha-dns-pair - configure a High Availability DNS relationship

### Synopsis

```
ha-dns-pair <name> create <main-cluster/address>
                        <backup-cluster/address>
                        [<attribute>=<value> ...]

ha-dns-pair <name> delete
ha-dns-pair list
ha-dns-pair listnames
ha-dns-pair <name> show

ha-dns-pair <name> get
<attribute>
ha-dns-pair <name> set
<attribute>=<value> [<attribute>=<value> ...]
ha-dns-pair <name> unset
<attribute>

ha-dns-pair <name> sync < update | complete | exact>
                    < main-to-backup | backup-to-main> [from-regional] >
```

### Description

The ha-dns-pair command lets you define and manage the High Availability relationship between a main and backup DNS server.

Either the main and backup clusters or the main and backup server IP addresses can be specified with the create command. If the ha-dns-main-server and ha-dns-backup-server addresses are set, the cluster addresses will only be used for synchronization of the server configuration. The referenced clusters must be configured with appropriate connection credentials for the sync command to be successful.

Note: When running in local mode, the 'from-regional' sync option does not apply. Regardless of the synchronization option (from-regional, main-to-backup, backup-to-main), properties set on the ha-dns-pair will always replace values present on the DNS server object.

### Examples

### Status

### See Also

[cluster](#)

### Attributes

**backup oid**(CCMCluster)

The cluster reference for the backup server in this DNS HA

pair relationship.

**ha-dns bool default = enabled**

This attribute enables/disables HA on the DNS server.

**ha-dns-backup-server ipaddr**

The IP address to use for the HA DNS protocol on the backup server. If this value is unset, the address specified for the main server will be used. In general, it should only be set if the server is configured with different interfaces for configuration management and update requests. The HA DNS protocol should always be configured with the interface used to service updates.

**ha-dns-main-server ipaddr**

The IP address to use for the HA DNS protocol on the main server. If this value is unset, the address specified for the backup server will be used. In general, it should only be set if the server is configured with different interfaces for configuration management and update requests. The HA DNS protocol should always be configured with the interface used to service updates.

**main oid(CCMCluster)**

The cluster reference for the main server in this DNS HA pair relationship.

**name string required,unique**

The name of the DNS HA pair relationship.

## help

help - provides online help

### Synopsis

```
help
help <cmd> [<section> ...]
```

### Description

The help command provides online help.

If you type help without arguments, it displays a list of commands. If you type help with an argument, it displays the man page information for the command with that name.

You can select the sections of the man page output by specifying the section names after the 'help <cmd>' command. The section names are:

SYNOPSIS	the valid syntax for the command
DESCRIPTION	a textual description of the command behavior
EXAMPLES	examples of using the command
PROPERTIES	description of the attributes associated with the command
STATUS	description of the status codes returned by this command

### Examples

```
nrcmd> help
100 Ok
... displays the list of commands
```

```
nrcmd> help dns
100 Ok
... displays the contents of the dns page
```

```
nrcmd> help dns synopsis
100 Ok
SYNOPSIS
dns
help <cmd> [<section> ...]
```

## Status

## See Also

# import

`import` - loads server configuration information from a file

## Synopsis

```
import keys <file>
import leases <file>
import named.boot <file>
import named.conf <file>
import option-set <file>
```

## Description

The `import` command lets you import lease information into the DHCP server configuration or BIND configuration information into the DNS server configuration.

**import leases** <file>

Before you can import leases, you need to perform several configuration steps:

1. Configure scopes in the DHCP server for the leases that are going to be imported. (see the `scope` command.)
2. If the host names for the leases are going to be dynamically entered into DNS as part of the import, configure zones in the DNS server to allow dynamic updates. (see the `zone` command)
3. Set the DHCP server to import mode so that it will not respond to other lease requests during the lease importing. (see the `dhcp` command)

After the leases have been imported, take the DHCP server out of import mode so that it will respond to other lease requests.

**import named.boot** <file>

Imports an existing BIND 4.x.x configuration into DNS by parsing the BIND `named.boot` file and reading the zone data from the associated BIND zone files.

**import named.conf** <file> <protected | unprotected>

Imports an existing BIND 8 or BIND 9 configuration into DNS by parsing the BIND `named.conf` file and reading the zone data from the associated BIND zone files. If no `name-protection` option is specified, the default is protected.

**import keys**

Imports TSIG keys into the Cluster configuration by reading in key data from a file. This file can be generated by running the key generator utility (cnr\_keygen). The keys are written in BIND syntax and therefore can also be copied from a valid BIND configuration.

**import option-set**

Imports an option-set from specified file. Compatible files can be generated using the 'export option-set' command.

## Examples

## Status

## See Also

# key

key - Manage TSIG key objects

## Synopsis

```
key list
key listnames
key <name> show
key <name> create <secret> [<attribute>=<value>...]
key <name> delete
key <name> get <attribute>
key <name> set <attribute>=<value> [<attribute>=<value> ...]
key <name> unset <attribute>
```

## Description

The key command creates and manages transaction signature (TSIG) keys for DNS updates, zone transfers, queries, and recursions. TSIG security, as defined in RFC 2845, enables both DNS and DHCP servers to authenticate DNS updates. TSIG security uses the HMAC-MD5 (or keyed MD5) algorithm to generate a signature that is used to authenticate the requests and responses. The DHCP server uses TSIG keys to create TSIG resource records while processing DNS updates.

To configure TSIG security on a DHCP server, you must first create a shared key, then enable DNS update for your scopes by setting the dynamic-dns attribute to update-all). Also, enable the dynamic-dns-tsig attribute for forward or reverse zones for the scope or on the server level.

## Examples

## Status

## See Also

## Attributes

**algorithm** [enumstr](#)(hmac-md5=1) default = hmac-md5

The algorithm that this key is used with. Currently we only support hmac-md5.

**id** [int](#)

Displays an integer id for the key.

**secret** [key](#) required

A base64 encoded string used for transaction authentication.

**security-type** [enumstr](#)(TSIG=1) default = TSIG

The type of security that this key is going to be used for. Currently we only support TSIG keys.

**time-skew** [rangetime](#)(1s-60m) default = 5m

The time stamp fudge factor (amount that the time values can differ).

## ldap

ldap - Specifies the LDAP remote server's properties

### Synopsis

```
ldap list
ldap listnames
ldap <name> create <hostname> [<attribute>=<value>...]
ldap <name> delete
ldap <name> get <attribute>
ldap <name> set <attribute>=<value> [<attribute>=<value> ...]
ldap <name> unset <attribute>

ldap <name> disable <attribute>
ldap <name> enable <attribute>
ldap <name> show

ldap <name> setEntry <dictionary> <key>=<value>
ldap <name> getEntry <dictionary> <key>
ldap <name> unsetEntry <dictionary> <key>
```

### Description

The ldap command configures the LDAP servers that the DHCP server should communicate with. The DHCP server can read client configuration information from or write lease information to an LDAP enabled directory.

Use the setEntry, getEntry, and unsetEntry commands to set, query, and clear elements of the various dictionary properties in the LDAP server configuration. These dictionary properties provide a convenient mapping from strings keys to string values.

The dictionary values for the setEntry command are:

```
create-dictionary
create-string-dictionary
env-dictionary
```

query-dictionary  
update-dictionary

## Examples

## Status

## See Also

## Attributes

### **can-create** [bool](#) default = disabled

Controls whether a particular LDAP server can create new entries to use to store lease state updates. See the create properties: [create-dictionary](#), [create-string-dictionary](#), [dn-create-filter](#), and [create-object-classes](#).

### **can-query** [bool](#) default = disabled

Controls whether a particular LDAP server can be used for queries. See the query properties: [env-dictionary](#), [query-dictionary](#), [search-attribute](#), [search-filter](#), [search-scope](#), and [search-scope](#).

### **can-update** [bool](#) default = disabled

Controls whether a particular LDAP server can be used to store lease state updates. See the update properties: [update-dictionary](#), [update-search-attribute](#), [update-search-filter](#), [update-search-scope](#), and [update-search-scope](#).

### **connections** [int](#) default = 1

Determines the number of connections that the server can handle for an LDAP object. Network Registrar creates one thread for each connection in an LDAP object, and each thread can have a maximum of 10 requests associated with its request queue. This is primarily a performance-tuning attribute. In some cases, having more than one connection can improve overall throughput.

### **create-dictionary** [string](#)

Maps LDAP attributes to DHCP lease attributes. If an entry does not exist and needs to be created, entries in this dictionary are set to the value of its corresponding DHCP lease attribute.

### **create-object-classes** [string](#)

If [can-create](#) is enabled, specifies the object classes for which a new entry inherits.

### **create-string-dictionary** [string](#)

Maps LDAP attributes to user specified strings. If an entry does not exist and needs to be created, entries in this dictionary are set to the matching string.

### **default-attribute-value** [string](#) default =

Provides a default attribute value (a string) to insert into an attribute whose associated lease attribute values are not set in the lease.

### **dn-attribute** [string](#)

Determines how the server constructs the distinguished name.

of the LDAP entry to update or create.

If the server can use one of the lease attributes, it for specified dn-attribute using the dn-format string to con the object filter that specifies the LDAP server to modif

**dn-create-format** [string](#)

Provides the distinguished name (DN) for entry creation. required at the entry level and is replaced by the value dn-attribute. If you can construct the DN of the LDAP obj from one of the lease?s attributes, the server formats t dn-attribute using the dn-format string.

**dn-format** [string](#)

If the DN of the ldap object that is to be updated can be constructed from one of the lease's attributes, the speci dn-attribute will be formatted using the dn-format string construct the query filter.

**enabled** [bool](#) default = true

Enables or disables this LDAP remote server. Prevents DHCP from attempting to use an LDAP server that known to be unavailable.

**env-dictionary** [string](#)

Specifies the environment dictionary that allows the serv retrieve additional LDAP attributes along with client-ent attributes. If any of these are present in a query's resu their values are made available to scripts through the re environment dictionary. The LDA{ value is keyed by the va the LDAP query env-dictionary.

**hostname** [string](#) required

Sets the hostname of the server to connect to. LDAP serve hostnames.

**limit-requests** [bool](#) default = enabled

Controls whether there should be a limit on the number of outstanding queries on each LDAP client connection. See t property: max-requests.

**max-referrals** [int](#) default = 0

Limits the number of LDAP referrals the server follows wh querying. A value of zero prohibits following referrals.

**max-requests** [int](#) default = 20

Controls the number of outstanding queries. If you have set the 'limit-requests' feature to TRUE, any LDAP connection will limit the number of outstanding quer 'max-requests.' You can improve performance by limiting t number of outstanding queries.

**name** [string](#) required,unique

An arbitrary name used to refer to an individual server.

**password** [string](#)

Sets the password of a user with access to the parts of t that DHCP uses. Because you can configure LDAP servers to anonymous access, this is optional.

**port** [int](#)

Specifies the port on the remote server to connect to.

**preference** [int](#) default = 1

Specifies the preference order in which LDAP servers are A positive integer greater than or equal to one. One (1)

highest preference value.

**query-dictionary** [string](#)

Maps LDAP attributes and DHCP attribute names. The server to retrieve all LDAP attributes specified in the dictionary query succeeds, the values for any ldap attributes that are set in the corresponding client-entry attribute.

**query-timeout** [time](#) default = 3s

Specifies the number of seconds the DHCP server waits for to individual LDAP Query requests. After a query request the DHCP server will drop the request and not process it another LDAP connection or LDAP Server. A query-timeout v seconds is a good value.

Note: The timeout attribute configures the timeout for LD Update and Create requests.

**referral-attr** [string](#)

Indicates whether an LDAP response is a referral. The referral may not contain the DN for this query. If the DN is present (default), the server uses it as the search path, along with wildcard search-scope in the query that follows the referral the server builds the search path by formatting the data referral attribute with the referral-filter, using the existing search scope.

**referral-filter** [string](#)

In the absence of a distinguished name (DN), controls how server formats referral-attr data. In such cases, the server formats the referral attribute's data with this filter expression search path that uses the existing search-scope for the LDAP

**search-filter** [string](#)

Specifies the filter to apply in the client-entry query. formats the client's MAC address using the filter to specify object that contains the client-entry data.

**search-path** [string](#)

Designates an object in the directory to use as a query starting-point. Together, the path and the search-scope portion of the directory that the server will search.

**search-scope** [enumint](#)(BASE=0, ONELEVEL=1, SUBTREE=2) default = SUB

Controls the comprehensiveness of a search:

If you specify the scope to be SUBTREE, the server searches the children of the searchpath.

If you specify the scope to be ONELEVEL, the server searches the immediate children of the base object.

If you specify the scope to be BASE, the server searches base object itself.

**threadwaittime** [mstime](#) default = 100

Sets the number of milliseconds that an LDAP client connection polls for the results of outstanding queries or updates.

**timeout** [time](#) default = 10s

Controls the number of seconds the DHCP server waits for to an individual LDAP update or create request. If an LDAP request times out, the DHCP server resubmits it to other LDAP connections. Further, if the DHCP server receives no response (that is for an LDAP update or create) from an LDAP connection for seconds, DHCP marks this LDAP connection as 'Inactive' and the connection, then reconnects. A timeout value of 10 seconds is a good value for LDAP create and update operations.

Note: You can configure a separate timeout for LDAP query using the query-timeout attribute.

### **update-dictionary** *string*

Maps LDAP attributes to DHCP lease attributes. When an L is modified, each LDAP attribute that is present in this is set to the value of its corresponding DHCP lease attri

### **update-search-attribute** *string*

If the DN of the object to be updated cannot be determine directly, the DHCP server must issue a query to retrieve In that case, the DHCP server uses data in the lease's 'search-attribute,' and formats it using the 'update-search-filter' expression.

### **update-search-filter** *string*

If the DN of the object to be updated cannot be determine directly, the DHCP server must issue a query to retrieve In that case, the DHCP server uses data in the lease's 'search-attribute,' and formats it using the 'update-search-filter' expression.

### **update-search-path** *string*

Determines the starting point for the portion of the dire containing LDAP objects for the server to update.

### **update-search-scope** *enumint*(BASE=0, ONELEVEL=1, SUBTREE=2)

With update-search-path, controls the portion of the dire that contains the objects to be updated.

The scope can be SUBTREE (includes all children of the searchpath), ONELEVEL (includes only the immediate childr base object), or BASE (includes only the base object itse

### **username** *string*

Designates a user with access to the parts of the directo uses. Because you can configure LDAP servers to allow ano access, this is optional.

## lease

lease - Manage DHCP lease objects

### Synopsis

```
lease list [-vpn=<vpn-name>] [-count-only]
lease list -macaddr <mac-address> [-vpn=<vpn-name>]
lease list -subnet <ip-address> <mask>
lease list -lansegment <ip-address> <mask>
lease [<vpn-name>/]<ip-address> [show]

lease [<vpn-name>/]<ip-address> get <attribute>

lease [<vpn-name>/]<ip-address> activate
lease [<vpn-name>/]<ip-address> deactivate
lease [<vpn-name>/]<ip-address> force-available

lease [<vpn-name>/]<ip-address> macaddr
lease [<vpn-name>/]<ip-address> get-scope-name

lease [<vpn-name>/]<ip-address> send-reservation
lease [<vpn-name>/]<ip-address> delete-reservation
```

### Description

The lease command lets you view and manipulate current DHCP leases in the cluster.

When you specify the lease on which one of these commands is to operate, you may optionally specify a <vpn-name> in which the <ip-address> is to be found. You may specify the name of a currently defined vpn as the <vpn-name>, or use the reserved vpn name "global" (without the quotes) to specify the operation on leases which are not in any explicitly defined vpn. If you do not specify a <vpn-name>, then the session's current-vpn is used as a default.

```
lease list [-vpn=<vpn-name>] [-count-only]
lease list -macaddr <mac-address> [-vpn=<vpn-name>]
lease list -subnet <ip-address> <mask>
lease list -lansegment <ip-address> <mask>
lease [<vpn-name>/]<ip-address> [show]
```

The list command lists leases in the DHCP server. Only the leases in the current-vpn or specified vpn-name are listed. <vpn-name> may be "all" (without the quotes) to request leases in all vpns. If -count-only is specified, only the count of the number of leases is returned (no leases are displayed).

The list -subnet command lists all leases in a subnet (scopes whose address and mask match the query).

The list -lansegment command lists all leases in a LAN segment, meaning all leases in scopes whose address and mask match the query, as well as leases in secondary scopes whose primary scope's address and mask match the query.

The list -macaddr command lists all leases that are associated with the specified MAC address.

Note: The list -macaddr command for Network Registrar 6.3 and earlier clusters can be extremely slow. In release 7.0 and later, performance is improved. The recommended syntax is -macaddr=<mac-address>.

```
lease [<vpn-name>/]<ip-address> activate
lease [<vpn-name>/]<ip-address> deactivate
```

The activate and deactivate commands tell the DHCP server to make the specified lease active or inactive. An inactive lease is not given out, even if it is in the available state. Making a currently leased lease inactive does not affect its behavior until it has expired and become available again.

```
lease [<vpn-name>/]<ip-address> force-available
```

The force-available command forces the specified lease into the available state.

```
lease [<vpn-name>/]<ip-address> macaddr
```

The macaddr command provides the MAC address associated with the specified lease.

```
lease [<vpn-name>/]<ip-address> get-scope-name
```

The get-scope-name command provides the scope to which the lease belongs.

```
lease [<vpn-name>/]<ip-address> send-reservation
```

The now-deprecated `send-reservation` command sends an existing reservation immediately to the DHCP server without requiring a server reload. You must first run the `scope <scope-name> addReservation` command for this reservation.

The recommended sequence is:

```
nrcmd> scope <scope-name> addReservation <ip-address> <mac-address>
nrcmd> save
nrcmd> lease <ip-address> send-reservation
```

This sequence ensures that the configuration database contains the lease and reservation as well as the running server, since the `send-reservation` command directly affects only the running server's in-memory database, and does not itself persist past the next server reload.

#### **lease** [<vpn-name>/]<ip-address> **delete-reservation**

The now-deprecated `delete-reservation` command deletes an existing reservation immediately from the DHCP server without requiring a server reload. It is important to ensure that the configuration which the DHCP server uses for the next server reload also has the reservation removed. The recommended sequence is:

```
nrcmd> lease <ip-address> delete-reservation
nrcmd> scope <scope-name> removeReservation <ip-address>
nrcmd> save
```

Note: The `send-reservation` and `delete-reservation` commands are deprecated. Use synchronous `scope-edit-mode` instead. These commands are provided for command-syntax compatibility with prior versions and only support mac-address based reservations.

## Examples

## Status

## See Also

[session](#) `current-vpn`

## Attributes

### **address** [ipaddr](#)

Specifies the IP address of the lease. The address is added at creation.

### **binding-end-time** [date](#)

Within the DHCP lease database, holds the time at which a lease binding ended.

### **binding-start-time** [date](#)

Within the DHCP lease database, shows the time at which a lease binding began.

### **client-binary-client-id** [blob](#)

Displays the binary form of the client's client-identifier.

### **client-dns-name** [string](#)

Displays the client DNS name, which the DHCP server attempted (possibly successfully) to enter into the DNS server for a specified client.

This attribute is related to the client-host-name, but may be identical due to name collisions in the DNS server database.

### **client-domain-name** [string](#)

Displays the domain (if any) to which the client's DNS name is related.

### **client-flags** [flags](#)(client-valid=1, client-id-created-from-mac-address=2, client-id-up-to-date=3, client-id-up-to-date-in-mcd=4, reverse-dns-up-to-date=5, dns-update-pending=9, client-fqdn-present=10, client-updates-name=11, clear-host-name=12, host-name-has-changed=6, domain-name-has-changed=8, use-test-before-update=13, dual-zone-dns-update=14, client-invalid-due-to-macaddress=15, in-limitation-list=16, used-over-limit-client-class=17, synthesized-dns-name=18, reservation-uses-client-id=19, client-id-from-override-id=20, client-id-from-string=21)

Displays any of the following values associated with the current lease state:

- 2 client-id-created-from-mac-address  
Indicates that the client-id was created for internal use from the client supplied MAC address. It is never reported externally if this is true.
- 3 client-dns-name-up-to-date  
Indicates that the client-dns-name (A) is actually present in the DNS server database.
- 5 reverse-dns-up-to-date  
Indicates that the reverse (PTR) DNS entry is current in the DNS database.
- 9 dns-update-pending  
Indicates that a DNS operation is pending for this lease.
- 16 in-limitation-list  
Indicates that this lease is presently in a limitation period, using the limitation-id shown.

These are for internal use only:

```
client-valid
client-fqdn-present
client-updates-name
clear-host-name
host-name-has-changed
domain-name-has-changed
use-test-before-update
avoid-dns-retry
dual-zone-dns-update
client-invalid-due-to-macaddress
used-over-limit-client-class
synthesize-dns-name
reservation-uses-client-id
client-id-from-override-id
client-id-from-string
```

### **client-host-name** [string](#)

Displays the DNS name that the client requested the DHCP server to place in the DNS server.

### **client-last-transaction-time** [date](#)

Displays the time when the client most recently contacted the DHCP server.

### **client-mac-addr** [macaddr](#)

Displays the MAC address which the client presented to the server.

### **client-os-type** [string](#)

Indicates the operating system of the client. This attribute is only used by the updateSms keyword and has no other purpose. If enable failover, the main server transmits this value to the backup server. The syntax of this attribute's value is OS-name m. Operating system values are as follows:

- Microsoft Windows NT Server
- Microsoft Windows NT Advanced Server
- Microsoft Windows NT Workstation 4.0
- Microsoft Windows NT Workstation 3.51
- Microsoft Windows 2000 Professional
- Microsoft Windows 95
- Microsoft Windows 9x
- Microsoft Windows for Workgroups
- Microsoft Windows
- Dos
- Netware
- LANMAN Workstation
- LANMAN Server
- OS/2
- MAC OS

### **client-override-client-id** [blob](#)

The value of the override-client-id expression for this client. If it appears, it is used as the client-id for this client.

### **client-vendor-class** [option](#)

The most recently received client vendor class option.

### **client-vendor-info** [option](#)

The most recently received vendor-specific information option.

### **data-source** [enumint](#)(unknown=0, main-main=4, backup-main=5, main-backup=6, backup=7)

Records the original source of the lease data and the machine from which the data was retrieved.

- 0 unknown
- 4 main-main  
Indicates the data originated on the main server and was retrieved from the main server.
- 5 backup-main  
Indicates the data originated on the backup server and was retrieved from the main server.
- 6 main-backup  
Indicates the data originated on the main server and was retrieved from the backup server.
- 7 backup-backup  
Indicates the data originated on the backup server and was retrieved from the backup.

When viewing leases with the UI's, you will see all four values routinely, especially if load-balancing is enabled. When looking at lease history records, main-main and backup-backup are the usual values, but in cases where the lease history poller has determined that some data may be missing, then main-backup and backup-main can appear as well.

### **expiration date**

Displays the date and time the lease will expire.

### **flags flags**(reserved=1, valid=2, deactivated=3, initialized=4, failover-updated=not\_in\_ranges=6, dynamic=7, backup=8)

Displays flags that describe this lease:

1 reserved

The lease is reserved for some MAC address. The table that relates MAC addresses to leases is in the scope of this lease.

3 deactivated

The lease is deactivated, which means that it should not be used. Any client which is using a deactivated lease will be NAK'ed on their next renewal.

7 dynamic

Last written by server which knew only about the lease because it was created by a send-reservation command.

8 backup

Indicates that the state for this lease was received from a server whose role was backup with respect to this lease.

For internal use only:

initialized

valid

failover-updated

### **fwd-dns-update-config-name nameref**(DnsUpdateConfig)

Names the Dns update configuration object used to perform dynamic DNS update on a forward zone.

### **giaddr ipaddr**

If present, the contents of the last received non-zero giaddr field. This represents the relay agent through which the client and server last communicated.

### **lease-renewal-time date**

Displays the earliest time the client is expected to issue a renewal request.

### **limitation-id blob**

Displays the value set for a client-class or client limitation number of simultaneous active leases a DHCP server can give to devices on customer premises.

### **relay-agent-auth blob**

The contents of the 'authentication' suboption 8 of the relay-agent information option 82 from this client.

### **relay-agent-circuit-id blob**

Displays the circuit-id sub-option of the DHCP relay-agent information option 82 from this client.

### **relay-agent-device-class int**

The contents of the 'device-class' suboption 4 of the relay-agent information option 82 from this client.

### **relay-agent-option option**

Displays the contents of the relay-agent information option 82 from the most recent client interaction.

### **relay-agent-radius-class string**

Displays the contents, if any, of the RADIUS class attribute that was contained in the RADIUS Attributes suboption of the relay-agent information option 82 from this client.

### **relay-agent-radius-options blob**

The contents of the 'radius' suboption 7 of the relay-agent information option 82 from this client.

This suboption has additional structure that is available other attributes of this class.

**relay-agent-radius-pool-name** [string](#)

Displays the contents, if any, of the RADIUS framed-pool : 88 contained in the RADIUS attributes suboption of the DHCP relay-agent information option 82 from this client.

**relay-agent-radius-session-timeout** [int](#)

If present, the contents of the RADIUS 'session-timeout' : 27 that was contained in the RADIUS Attributes suboption ' relay-agent information option 82 from this client.

**relay-agent-radius-user** [string](#)

Displays the contents, if any, of the RADIUS user attribute contained in the RADIUS attributes suboption of the DHCP : information option 82 from this client.

**relay-agent-radius-v6-pool-name** [string](#)

If present, the contents of the RADIUS 'v6-pool-name' attribute that was contained in the RADIUS Attributes suboption 7 of relay-agent information option 82 from this client.

**relay-agent-radius-vendor-specific** [blob](#)

If present, the contents of the RADIUS 'vendor-specific' : 26 that was contained in the RADIUS Attributes suboption ' relay-agent information option 82 from this client.

**relay-agent-remote-id** [blob](#)

Displays the remote-id sub-option of the DHCP relay-agent information option 82 from this client.

**relay-agent-server-id-override** [ipaddr](#)

Displays the IP address in the server-id-override sub-option of the DHCP relay-agent information option 82 from this client. This value corresponds to one of two suboption numbers: If the IANA assigned suboption 182 is present in the packet, that value is returned; otherwise, if the Cisco suboption is present, that value is returned.

**relay-agent-subnet-selection** [ipaddr](#)

Displays the IP address in the subnet selection sub-option of the DHCP relay-agent information option 82 from this client. This value corresponds to one of two suboption numbers: If the IANA assigned suboption is present in the packet, that value is returned; otherwise, if Cisco suboption 15 is present, that value is returned.

**relay-agent-subscriber-id** [string](#)

Displays the contents of the subscriber-id suboption of the relay-agent information option 82 from this client.

**relay-agent-v-i-vendor-class** [blob](#)

The contents of the 'v-i-vendor-class' suboption 9 of the relay-agent information option 82 from this client.

**relay-agent-vpn-id** [blob](#)

Displays the contents of the vpn-id sub-option of the DHCP relay-agent information option 82 from this client. This value corresponds to one of two suboption numbers: If the IANA assigned suboption 181 is present in the packet, that value is returned; otherwise, if Cisco suboption 151 is present, that value is returned.

**reservation-lookup-key blob**

The lookup key with which a reservation for this lease will be retrieved.

**rev-dns-update-config-name nameref(DnsUpdateConfig)**

Names the Dns update configuration object used to perform dynamic DNS update on a reverse zone.

**scope-name nameref(Scope)**

A reference to the scope that contains this lease.

**start-time-of-state date**

Displays the time the state changed to its current value.

**state enumint(available=1, offered=2, leased=3, expired=4, unavailable=5, released=6, other-available=7, pending-available=8)**

Displays the current state of the lease.

- 1 available  
The lease is not currently leased by any client. Any client information is from the most recent client to lease or be offered this lease.
- 2 offered  
The lease is offered to the associated client. In many cases, the database is not written with information concerning offering a lease to a client since there is no requirement to update stable storage with this information.
- 3 leased  
The lease is currently leased to the client whose information appears in the lease.
- 4 expired  
The client specified has not renewed the lease, and it has expired. Upon expiration the DNS information for the client was scheduled for removal.
- 5 unavailable  
The lease is unavailable. It was made unavailable because of some conflict. A ping attempt might have shown that another client using the, or the DHCP server might have detected another DHCP server handing out this IP address, or a DHCP client might have declined the lease. Use start-time-of-state to determine when the lease became unavailable, and look in the log file around that time to determine exactly why the lease became unavailable.
- 6 released  
The client specified has released the lease, but the server was configured to apply a 'release-grace-period'. The lease won't be made available until the grace-period expires.
- 7 other-available  
Used only when failover is enabled. A lease in the other-available state is available for allocation by the other server in the failover pair, but not available for allocation by this server.
- 8 pending-available  
Used only when failover is enabled. A lease in the pending-available state will be available as soon as this server can synchronize its available state with the other server.

**user-defined-data string**

This string value is associated with the lease in order to

customer applications to relate the lease record to other databases. It is not used directly by the DHCP server, but read and written by extensions and expressions.

**vendor-class-id** *string*

Displays the vendor-class-id as offered in a DHCP request

**vpn-id** *int* default = 0

Displays the identifier of the DHCP VPN that contains this

## lease-notification

lease-notification - Reports scopes with few free leases

### Synopsis

```
lease-notification available=<number>|<percentage>
[config=config file]
[leasing-only]
[scopes=<scope name>|<address range>
[,<scope name>|<address range>,...]]
[[recipients=<recipient>[,<recipient>,...]]
[mail-host=<name> [errors-to=<recipient>]] ]
[vpn=<vpn-name>]
```

### Description

Use the lease-notification command to receive notification about the number of available addresses in a scope. This command reports on the scopes for which the number of available addresses falls below or equals a set value. You can specify the notification limit either as the number of free addresses or the percentage of free addresses. You can also specify who should receive e-mail notification.

Although you can use the lease-notification command interactively, its primary use is as an automated command.

You can specify clusters in several ways:

- The default cluster (localhost)
- The AIC\_CLUSTER environment variable or NT registry entry
- The -C flag on the command line
- The clusters property in the config file lets you specify a group of clusters. For example, to specify several clusters, enter the following in the config file:

```
# Cluster information for summary reports
[lease-notification]
# Clustername Username Password
clusters=host1 admin passwd1, host2 admin2,host3,
        host4 admin4 passwd4
```

Follow these guidelines for specifying clusters:

- Separate cluster specifications from each other with commas.
- Separate arguments for a particular cluster by whitespace.
- For long lines use continuation lines; you do not continuation escape indicators.

- Optionally, specify a user name and password for the cluster. If you do not provide a user name or password for a particular cluster, Network Registrar uses the last user name or password listed. If you do not provide user names or passwords, Network Registrar uses the information from the command line -N and -P arguments, and then the NT Registry or environment variables AIC\_NAME and AIC\_PASSWORD. If Network Registrar cannot find a user name or password, or the supplied user name and password are incorrect, the lease-notification command issues a warning for that cluster.

The lease-notification command output consists of an explanatory header, a table containing a row for each scope in which the number of free addresses is equal to or less than the threshold, and possible warnings related to the scopes and clusters requested.

The lease-notification command reports the following information for each scope in the table:

- Cluster name
- Scope name
- Scope network address in the canonical dotted format
- Number of high-order bits in the scope subnet mask
- Number of addresses in the scope ranges
- Percentage of addresses available for lease
- Number of addresses available for lease (addresses that are reserved or deactivated are not included as free)

## Lease Notification Keywords

### available

Specifies either a number or percentage of available addresses. If the number or percentage of available addresses is equal to or less than the specified value for the scopes being checked, Network Registrar generates a report listing information about the scopes that reach or exceed the available value.

### config

Specifies a configuration file. If you don't specify a configuration file, Network Registrar searches for the default .nrconfig file.

### errors-to

If you specify a mail-host, you may also specify the email address of the sender of the email in order to provide a return path for bounced email. The default value is "postmaster".

### leasing-only

Specifies that only scopes that can currently offer leases are reported.

### mail-host

On NT, you must specify a mail-host. On Solaris the mail host is generally already configured for the sendmail program. You can verify that your Solaris system is properly configured by issuing the command "date | mail <your-email-address>" and observing whether or not the date is emailed to you.

## recipients

If you specify the email addresses of one or more recipients, Network Registrar sends an email report to those addresses. Otherwise, Network Registrar directs the report to standard output.

## scopes

The scopes to check either by name or as a range or ranges of addresses. Network Registrar checks any scope containing any address that falls within a range of address. If you don't list any scopes or addresses, Network Registrar checks all scopes managed by the specified cluster.

## vpn

The VPN from which to select scopes to examine when executing this command. If no VPN name is specified, then the current VPN of the session is used. If the reserved VPN name "global" is used, then the global (or unnamed) VPN is used. If the reserved VPN name "all" is used, then all scopes from all vpns are examined.

## Examples

## Status

## See Also

[report](#), [export](#) addresses, [session](#) current-vpn

# lease6

lease6 - Manage DHCP lease6 objects

## Synopsis

```
lease6 list [-duid=<client-id>]
           [-lookup-key=<lookup-key> [-blob|-string]]
           [-macaddr=<mac-addr>] [-cm-macaddr=<mac-addr>]
           [-vpn=<vpn-name>] [-count-only]
lease6 [<vpn-name>/]<ipv6-address> [show]
lease6 [<vpn-name>/]<ipv6-address> get <attribute>
lease6 [<vpn-name>/]<ipv6-address> activate
lease6 [<vpn-name>/]<ipv6-address> deactivate
lease6 [<vpn-name>/]<ipv6-address> force-available
lease6 [<vpn-name>/]<ipv6-address> reconfigure
           [renew|rebind|information-request] [-unicast|-via-relay]
```

## Description

The lease6 command lets you view and manipulate the current DHCPv6 leases in the cluster.

When you specify the lease on which one of these commands is to operate, you may optionally specify a <vpn-name> in which the <ipv6-address> is to be found. Specify the name of a currently defined vpn as the <vpn-name>; or use the reserved vpn name "global" (without the quotation marks) to specify the operation

on leases which are not in any explicitly defined VPN. If you do not specify a vpn-name, the current VPN of the session is used.

```
lease6 list [-duid=<client-id>]
            [-lookup-key=<lookup-key> [-blob|-string]]
            [-macaddr=<mac-addr>] [-cm-macaddr=<mac-addr>]
            [-vpn=<vpn-name>] [-count-only]
```

The list command lists DHCPv6 leases in the DHCP server. Only the leases in the current VPN or specified vpn-name are listed. The vpn-name may be "all" (without the quotation marks) to request leases in all VPNs. If -count-only is specified, only the count of the number of leases is returned (no leases are displayed). If a filter (-duid, -lookup-key, -macaddr, or -cm-macaddr) is specified, only the leases matching the filter are displayed.

```
lease6 [<vpn-name>/]<ipv6-address> activate
lease6 [<vpn-name>/]<ipv6-address> deactivate
```

The activate and deactivate commands tell the DHCP server to make the specified lease active or inactive. An inactive lease is not given out, even if it is in the available state. Making a currently leased lease inactive will not affect its behavior until it has expired and become available again.

```
lease6 [<vpn-name>/]<ipv6-address> force-available
```

The force-available command forces the specified lease into the available state.

```
lease6 [<vpn-name>/]<ipv6-address> reconfigure
```

The reconfigure command initiates sending the client a Reconfigure message (if the client and server negotiated to allow reconfigure).

## Examples

## Status

## See Also

[session](#)

## license

license - Views and updates license information

## Synopsis

```
license <FLEXlm-filename> create
license <key> delete
license list
license listnames
license <key> [show]
license <key> get <attribute>
license showUtilization
```

## Description

The license command allows you to view, create, or delete the FLEXlm licenses for the cluster. The command (showUtilization) also allows you to view the number of utilized IP nodes against the RTU's (Right-to-Use).

NOTE: The license command uses a different syntax when connected to releases prior to 7.0. See the documentation for the specific release.

## Examples

## Status

## See Also

The "Network Registrar CLI Introduction" section describes how licenses are used in nrcmd.

## link

link - configures IPv6 network links for use in DHCPv6

Note: dhcp-link is a synonym for compatibility with earlier versions.

## Synopsis

```
link list
link listnames
link <name> create [[template-root-prefix=<prefix>]
                  template=<template-name>]
                  [attribute]=<value> ...]
link <name> delete
link <name> set =<value> [attribute]==<value> ...]
link <name> get <attribute>
link <name> unset <attribute>
link <name> enable <attribute>
link <name> disable <attribute>
link <name> [show]

link <name> listPrefixes
link <name> listPrefixNames

link <name> applyTemplate <template-name> [<template-root-prefix>]
```

## Description

The link command configures IPv6 network links. Links group IPv6 prefixes (see the prefix command) together. Links are required if multiple prefixes share the same physical link.

When creating a link using a template, specify - for the <name> to allow the link template's link-name-expr to name the link.

## Examples

## Status

## See Also

[link-template](#), [prefix](#)

## Attributes

### **description** [string](#)

Describes the link.

### **embedded-policy** [obj\(0\)](#)

Refers to a policy embedded within a single specific link object used when replying to clients.

### **free-address-config** [nameref\(AddrTrapConfig\)](#)

Identifies which trap captures unexpected free address event on this Link.

If this attribute is not configured, the server looks for the v6-default-free-address-config on the DHCPServer object.

### **local-cluster** [oid](#)

Identifies the local DHCP cluster or failover pair for this regional link.

### **name** [string](#) required,unique,immutable

Provides user-assigned name for the link.

### **policy** [nameref\(Policy\)](#) default = default

Refers to a shared policy used when replying to clients.

### **prefix-list** [obj\(Prefix\)](#) transient

Lists the prefixes to be associated with the link. This attribute is used to add or modify the link and its prefixes in a single database action. All objects must be valid, or none be accepted. The associated prefixes will be stored separately and will not be returned in the parent link object.

### **template-root-prefix** [prefix](#) transient

Identifies the root prefix address for prefixes associated with the link. This attribute is used when processing a link template that defines associated prefixes.

### **vpn-id** [int](#) default = 0, immutable

Identifies the VPN that contains the link.

## link-policy

link-policy - Edits a DHCP policy embedded in a link.

Note: dhcp-link-policy is a synonym for compatibility with earlier versions.

## Synopsis

```
link-policy <name> delete
link-policy <name> set <attribute>=<value>
                        [<attribute>=<value> ...]
link-policy <name> get <attribute>
link-policy <name> disable <attribute>
link-policy <name> enable <attribute>
link-policy <name> show
```

```

link-policy <name> setV6Option <opt-name | id> <value>
link-policy <name> getV6Option <opt-name | id>
link-policy <name> unsetV6Option <opt-name | id>
link-policy <name> listV6Options

link-policy <name> setV6VendorOption <opt-name | id>
                                <opt-set-name> <value>
link-policy <name> getV6VendorOption <opt-name | id>
                                <opt-set-name>
link-policy <name> unsetV6VendorOption <opt-name | id>
                                <opt-set-name>
link-policy <name> listV6VendorOptions

```

## Description

The link-policy command lets you configure a DHCP policy embedded in a DHCP link. An embedded policy is a collection of DHCP option values and settings that are associated with (and named by) another object -- in this case a link. You create a link-policy when you first reference it, and you delete it when you delete the link.

To set individual option values use the setV6Option command; to unset option values, the unsetV6Option command; and to view option values, the getV6Option and listV6Options commands. When you set an option value, the DHCP server replaces any existing value or creates a new one as needed for the given option name.

See the help file for the policy command for more information.

## Examples

## Status

## See Also

[policy](#), [client-policy](#), [client-class-policy](#), [dhcp-address-block-policy](#), [prefix-policy](#), [policy](#), [scope-template-policy](#)

## Attributes

### affinity-period **time**

Associates a lease in the AVAILABLE state with the client that last held the lease. If the client requests a lease during the affinity period, it is granted the same lease; that is, unless renewals are prohibited, then it is explicitly not given the lease. Because of the vast IPv6 address space and depending on the generation technique, it could be millions of years before the address ever needs reassignment to a different client, and there is no reason to hold on to this information for that long. To prohibit renewals enable either the inhibit-all-renewals attribute or the inhibit-renews-at-reboot attribute.

### allow-client-a-record-update **bool** default = disabled

Determines if a client is allowed to update A records. If the client sets the flags in the FQDN option to indicate that it wants to do the A record update in the request, and if the allow-client-a-record-update value is TRUE, the server allows the client to do the A record update; otherwise, based on other server configurations, the

does the A record update.

**allow-dual-zone-dns-update** **bool** default = disabled

Enables DHCP clients to perform DNS updates into two DNS zones. To support these clients, you can configure the DHCP server to allow the client to perform an update, but also to perform update on the client's behalf.

**allow-lease-time-override** **bool** default = disabled

Gives the server control over the lease period. Although a client can request a specific lease time, the server need not honor the request if this attribute is set to false (the default). Even if set to true, clients can request only lease times that are shorter than those configured for the server.

**allow-non-temporary-addresses** **bool** default = true

Determines whether DHCPv6 clients can request non-temporary (IA\_NA) addresses.

The default is to allow clients to request non-temporary addresses.

**allow-rapid-commit** **bool** default = false

Determines whether DHCPv6 clients can use a Solicit with the Rapid Commit option to obtain configuration information with fewer messages. To permit this, make sure that a single DHCP server is servicing clients.

This attribute has special handling during the policy hierarchy processing when checking the Prefix policies (embedded or not) for the Prefixes on a Link. The Prefixes for the Link are processed in alphabetic (case blind) order. For each Prefix, the embedded and then named policy are checked. Only Prefixes that the client has access (based on selection tags, etc.) are checked.

- If any of the prefix policies has this attribute set to FALSE, Rapid Commit is not allowed.
- If at least one has it set to TRUE, Rapid Commit is allowed.
- Otherwise, the remaining policies in the hierarchy are checked.

The default is not to allow clients to use Rapid Commit.

**allow-temporary-addresses** **bool** default = true

Determines whether DHCPv6 clients can request temporary (IA\_TA) addresses.

The default is to allow clients to request temporary addresses.

**default-prefix-length** **rangeint(0-128)** default = 64

For delegation, specifies the default length of the delegation if a router (client) does not explicitly request it.

The default length must always be less than or equal to the length of the prefix range.

**forward-dnsupdate** **nameref(DnsUpdateConfig)**

Specifies the name of the update configuration that determines which forward zones to include in updates.

**forward-zone-name** **dname**

Designates an optional forward zone for DNS updates.

**giaddr-as-server-id** **bool** default = false

Enables the DHCP server to set the server-id option on a Discover and a DHCPACK to the giaddr of the incoming packet, instead of the IP address of the server (the default action).

This causes all unicast renews to be sent to the relay agent or directly to the DHCP server, and so renews arrive at the server with option-82 information appended to the packet.

Some relay agents may not support this capability and, in that case, the server will not set the server-id option.

complex configurations, the giaddr might not actually be an address to which the DHCP client can send a unicast packet. In these cases, the DHCP client cannot renew a lease, and must always perform a rebinding operation (where the DHCP client broadcasts a request for sending a unicast to what it believes is the DHCP server).

**grace-period** *time* default = 5m

Defines the length of time between the expiration of a lease and the time it is made available for reassignment.

**inhibit-all-renews** *bool* default = false

Causes the server to reject all renewal requests, forcing clients to obtain a different address any time it contacts the DHCP server.

**inhibit-renews-at-reboot** *bool* default = false

Permits clients to renew their leases, but the server forces them to obtain new addresses each time they reboot.

**limitation-count** *int*

Specifies the maximum number of clients with the same limitation that are allowed to have currently active and valid leases.

**offer-timeout** *time* default = 2m

Instructs the server to wait a specified amount of time when it has offered a lease to a client, but the offer is not yet accepted. At the end of the specified time interval, the server makes the lease available again.

**packet-file-name** *string*

Identifies the boot-file to use in the boot process of a client. The server returns this file name in the 'file' field of its replies. The packet-file-name cannot be longer than 128 characters.

**packet-server-name** *string*

Identifies the host-name of the server to use in a client's boot process. The server returns this file name in the 'sname' field of its replies. The packet-server-name field cannot be longer than 64 characters.

**packet-siaddr** *ipaddr*

Identifies the IP address of the next server in the client's boot process. For example, this might be the address of a TFTP server used by BOOTP clients. The server returns this address in the 'siaddr' field of its replies.

**permanent-leases** *bool* default = disabled

Indicates whether leases using this policy are permanently granted to requesting clients. If leases are permanently granted, the server ignores the configured dhcp-lease-time option value and uses the maximum lease time.

**preferred-lifetime** *time* default = 1w

Assigns the default and maximum preferred lifetime for leases on DHCPv6 client interfaces. Expressed in seconds and relative to the time the server sent the packet, this attribute sets the length of time that the address is preferred; that is, its use is unrestricted. When the preferred lifetime expires, the address becomes deprecated and its use is restricted.

**reconfigure** *enumint*(allow=1, disallow=2, require=3) default = allow

Controls DHCPv6 client reconfiguration support:

1	allow	Allows clients to request reconfiguration support and the server will honor the request (default).
2	disallow	Allows clients to request reconfiguration support but the server will not honor the request.

- 3 require support but the server will not honor the clients' request.
- Requires clients to request reconfiguration support and the server drops client Solicit and Request messages that do not include a Reconfigure-Accept option.

This attribute has special handling during the policy hierarchy processing when checking the Prefix policies (embedded or named) for the Prefixes on a Link. The Prefixes for the Link are processed in alphabetic (case blind) order. For each Prefix embedded and then named policy are checked. Only Prefixes that the client has access (based on selection tags, etc.) are checked as follows:

- If any of the prefix policies has this attribute set to disallow or require, that setting is used.
- Otherwise, if at least one has it set to allow, Reconfigure is allowed.
- If no prefix policies have this attribute set, the remaining policies in the hierarchy are checked.

**reconfigure-via-relay** **bool** default = false

Controls whether the server should prefer unicasting or relaying DHCPv6 Reconfigure messages.

If false (the default), the server prefers to unicast Reconfigure messages if the client has one or more valid statefully assigned addresses.

If true, the server prefers to send Reconfigure messages via the relay agent unless no relay agent information is available.

Note: When you use this attribute, consider that:

- In networks where the DHCPv6 server cannot communicate directly with its client devices?for example, where firewalls are in place?set this value to true.
- The DHCPv6 server does not use embedded and named policies configured on a client when it evaluates this attribute.
- The relay agent cannot be used if the Relay-Forw message came from a link-local address.

**reverse-dnsupdate** **nameref**(DnsUpdateConfig)

Specifies the name of the update configuration that determines which reverse zones to include in a DNS update.

**server-lease-time** **time**

Tells the server how long a lease is valid. For more frequent communication with a client, you might have the server consider leases as leased for a longer period than the client considers. This also provides more lease-time stability. This value is used unless it is longer than the lease time in the dhcp-lease-time found through the normal traversal of policies.

**split-lease-times** **bool** default = disabled

Specifies a value that the DHCP server might use internally to affect lease times.

If enabled, the DHCP server still offers clients lease times that reflect the configured lease-time option from the appropriate policy; but the server bases its decisions regarding expiration on the 'server-lease-time' value.

**unavailable-timeout** **time** default = 24h

Permits the server to make a lease unavailable for the time specified and then to return the lease to available state. If there is no

configured in the `system_default_policy`, then the default is 86400 seconds (or 24 hours).

#### **use-client-id-for-reservations** `bool` default = off

Controls how the server database checks for reserved IP addresses.

By default, the server uses the MAC address of the DHCP client for its database lookup. If this attribute is set to `on` (enabled), then the server does the check for reserved addresses using the DHCP client-id, which the client usually supplies. In cases where the DHCP client does not supply the client-id, the server synthesizes it, and uses that value.

#### **v4-bootp-reply-options** `optionid4`

Lists the options the server returns to all BOOTP clients.

#### **v4-reply-options** `optionid4`

Lists the options the server returns to all DHCPv4 clients or not the client specifically asks for the option data.

#### **v6-reply-options** `optionid6`

Lists the options that should be returned in any replies to DHCPv6 clients.

This attribute has special handling during the policy hierarchy processing when checking the Prefix policies (embedded or not) for the Prefixes on a Link. The Prefixes for the Link are processed in alphabetic (case blind) order. For each Prefix, the embedded and then named policy are checked. Only Prefixes that the client has access (based on selection tags, etc.) are checked.

#### **valid-lifetime** `time` default = 2w

Assigns the default and maximum valid lifetime for leases to DHCPv6 client interfaces. Expressed in seconds and relative to the time the server sent the packet, this attribute sets the length of time that an address remains valid. When the period of time expires, the address becomes invalid and unusable. The valid lifetime must be greater than or equal to the preferred lifetime.

## link-template

`link-template` - Configures a link template

### Synopsis

```
link-template <name> create [<attribute>=<value>...]
link-template <name> delete
link-template <name> set <attribute>=<value> [<attribute>=<value> ...]
link-template <name> get <attribute>
link-template <name> unset <attribute>

link-template <name> disable <attribute>
link-template <name> enable <attribute>
link-template <name> show

link-template <name> create clone=<clone-name>
link-template <name> apply-to < all | <link1>[,...] >
link-template <name> apply-to <link> [<prefix>]
```

### Description

The `link-template` command lets you configure a template to use when creating links.

## Examples

## Status

## See Also

[link](#)

## Attributes

### **description** [string](#)

Describes the link template.

### **free-address-config** [nameref](#)(AddrTrapConfig)

Identifies which trap captures unexpected free address on this link.

If this attribute is not configured, the server looks for the v6-default-free-address-config on the DHCPServer object.

### **link-description-expr** [expr](#)

An expression to define the description on the link object created when using the template.

### **link-name-expr** [expr](#)

An expression to define the name of the link object created when using the template.

### **name** [string](#) required,unique,immutable

Assigns a name to this link template.

### **options-expr** [expr](#)

An expression to define the list of embedded policy options to be created.

### **policy** [nameref](#)(Policy) default = default

Refers to a shared policy used when replying to clients.

### **prefix-expr** [expr](#)

Defines an expression used to create the list of associated prefixes.

## option

option - Configures option definitions

### Synopsis

```
option <id> <option-setname> create <option-name> <type> [<attribute>=<value>]
```

```
option <name | id> <option-setname> delete
```

```
option <name | id> <option-setname> set <attribute>=<value> [<attribute>=<value>]
```

```
option <name | id> <option-setname> get <attribute>
```

```
option <name | id> <option-setname> [show]
```

```
option <name | id> <option-setname> unset <attribute>
```

```
option <name | id> <option-setname> enable <attribute>
```

```
option <name | id> <option-setname> disable <attribute>
```

```
option <option-setname> list
```

```
option <option-setname> listnames
```

```
option <option-setname> show
```

```
option listtypes
```

## Description

The option command configures option definitions.

Use reserved names as follows:

dhcp-config and dhcp6-config to view currently configured option sets for DHCPv4 and DHCPv6 respectively.

dhcp-custom and dhcp6-custom to view/add/modify/delete custom option definitions.

NOTE: you may also use dhcp-config and dhcp6-config to add/modify/delete custom option definitions. These names are used to operate on the respective custom set.

Changes to the custom sets are merged with the built-in option definitions to form the config sets.

Modifications to the custom sets are not visible in the config sets until after a SAVE is performed.

You cannot use the reserved names when creating or deleting an option set. The custom sets are created when the first custom option definition is created.

Use the listtypes command to view the list of option types available for use in creating custom option definitions.

## Examples

## Status

## See Also

## Attributes

### number [int](#)

The option number (the T part of the TLV)

### option-definition-set-name [nameref](#)(OptionDefinitionSet)

A reference to the option-definition-set used to create this option instance.

### option-desc [objref](#)(AttrDesc)

The option description that provides the type and name for this specific option. This is a shortcut for a lookup by id in the appropriate TLV desc table.

### sub-options [obj](#)(Option)

This attribute provides for the optional subdivision of option data into nested sub-option objects. This is not used for storing DHCP server configuration, but may be useful in other uses of TLV based values.

### value [blob](#)

The option value (the V part of the TLV)

## option-set

option-set - Configure option definition sets

### Synopsis

```
option-set <name> create <8-bit | 16-bit> vendor-option-string=<string> [<att
option-set <name> create <8-bit | 16-bit> vendor-option-enterprise-id=<intege
option-set <name> delete
option-set list
option-set listnames
option-set <name> [show]
option-set <name> set <attribute>=<value> [<attribute>=<value> ...]
option-set <name> get <attribute>
option-set <name> unset <attribute>

option-set <name> enable <attribute>
option-set <name> disable <attribute>

option-set dhcp-custom unset
option-set dhcp6-custom unset
```

### Description

The option-set command configures option definition sets.

Use reserved names as follows:

dhcp-config and dhcp6-config to view currently configured option sets for DHCPv4 and DHCPv6 respectively. These reserved names show all built-in and custom option definitions.

dhcp-custom and dhcp6-custom to view only custom option definitions.

Changes to the custom sets are merged with the built-in option definitions to form config sets.

Modifications to the custom sets are not visible in the config sets until after you perform a SAVE.

You cannot use the reserved names when creating or deleting an option set. The custom sets are created when the first custom option definition is created. Use the command unset if you wish to clear all custom option definitions.

Use 8-bit to create a dhcpv4 vendor option definition set.  
Use 16-bit to create a dhcpv6 vendor option definition set.

### Examples

### Status

### See Also

# owner

owner - Configures owners

## Synopsis

```
owner <tag> create <name> [<attribute>=<value>]
owner <tag> delete
owner list
owner listnames
owner <tag> show
owner <tag> set =<value> [<attribute>=<value> ...]
owner <tag> get <attribute>
owner <tag> unset <attribute>

owner <tag> enable <attribute>
owner <tag> disable <attribute>
```

## Description

The owner command configures a specified owner. You create owners and associate them with address blocks, subnets, and zones.

## Examples

## Status

## See Also

## Attributes

### contact [string](#)

Provides contact information for this owner.

### name [string](#) required

Displays the full name, or printable name, for this owner.

### organization [nameref\(CCMOrganization\)](#)

Specifies the organization name required for ARIN reporting

### tag [string](#) required,unique

Displays a unique tag name for this owner. Typically, it is name referring to this owner.

# prefix

prefix - Configures IPv6 network prefixes for use in DHCPv6

Note: dhcp-prefix is a synonym for compatibility with earlier versions of Network Registrar.

## Synopsis

```
prefix list
prefix listnames
prefix <name> create <address> [template=<template-name>]
                        [<attribute>=<value>]
prefix <name> delete
```

```

prefix <name> set <attribute>=<value> [<attribute>=<value> ...]
prefix <name> get <attribute>
prefix <name> unset <attribute>
prefix <name> enable <attribute>
prefix <name> disable <attribute>
prefix <name> [show]

prefix <name> listLeases

prefix <name> addReservation <address> <lookup key>
prefix <name> removeReservation <address>
prefix <name> listReservations

prefix <name> applyTemplate <template-name>

```

## Description

The `prefix` command configures IPv6 network prefixes. Prefixes configure DHCPv6 address allocation and prefix delegation.

When creating a prefix using a template, specify `-` for the `<name>` to allow the prefix template's `prefix-name-expr` to name the prefix.

## Examples

```

nrcmd> prefix example-pref create ff00::/8
nrcmd> prefix example-pref set address=ff00::/10

```

## Status

## See Also

[link](#), [prefix-template](#)

## Attributes

**address** [prefix](#) required,immutable

Identifies a prefix (subnet) that an interface belongs to using high-order bits of an IPv6 address.

**allocation-algorithms** [flags](#)(client-request=1, reservation=2, extension=3, interface-identifier=4, random=5, best-fit=6) default = reservation,extension,random,best-fit

Controls the algorithms used by the server to select a new address or prefix to lease to a client. The available algorithms are:

`client-request`

This setting (off by default) controls whether the server uses a client requested lease.

`reservation`

This setting (on by default) controls whether the server uses an available reservation for the client.

`extension`

This setting (on by default) controls whether the server extensions attached at the generate-lease extension point to generate an address or prefix for the client.

`interface-identifier`

This setting (off by default) controls whether the server uses the interface-identifier from the client's (link-local)

address to generate an address. It is ignored for temporary addresses and prefix delegation.

random

This setting (on by default) controls whether the server generates an address using an RFC 3041-like algorithm. It is ignored for prefix delegation.

best-fit

This setting (on by default) controls whether the server delegate the first, best-fit available prefix. It is ignored for addresses.

When the server needs an address to assign to a client, it processes the flags in the following order: client-request, reservation, extension, interface-identifier, and random. Processing stops when a usable address is produced.

When the server needs to delegate a prefix to a client, it processes the flags in the following order: client-request, reservation, extension, best-fit. Processing stops when a usable prefix is produced.

### **deactivated** **bool** default = disabled

Controls whether a prefix extends leases to clients. A deactivated prefix does not extend leases to any clients. It treats all addresses in its ranges as if they were individually deactivated. Default, false (active).

### **description** **string**

Describes the prefix.

### **dhcp-type** **enumint**(stateless=0, dhcp=1, prefix-delegation=2, infrastructure=3, parent=4) default = dhcp

Defines how DHCP manages address assignment within a prefix:

stateless	Prefix is used only for stateless option configuration.
dhcp	Prefix is used for stateful address assignment.
prefix-delegation	Prefix is used for prefix-delegation.
infrastructure	Prefix is used to map a client address to a link, and does not have an address pool.
parent	Prefix is not used by DHCP. It is used as a container object, to group child prefixes.

### **embedded-policy** **obj**(0)

Specifies a policy embedded within an IPv6 prefix.

### **expiration-time** **date**

Sets the time and date on which a prefix expires. After this time and date, the server neither grants new leases nor renews existing leases from this prefix.

Once the expiration-time has passed, the prefix is no longer available (though old leases and leases with grace or affinity periods continue to exist until those periods elapse).

Enter this as a date in the format "[weekday] mon day hh:mm[:ss] year". For example, "Dec 31 23:59 2006".

### **free-address-config** **nameref**(AddrTrapConfig)

Identifies which trap captures unexpected free address events on this prefix.

If this attribute is not configured, the server looks for the free-address-config attribute on the parent Link object.

If that attribute is not configured, the server looks for the v6-default-free-address-config on the DHCP Server object.

**ignore-declines** `bool` default = false

Controls whether the DHCP server responds to a DHCPv6 DECLINE message that refers to an IPv6 address or a delegated prefix from this prefix.

If enabled, the DHCP server ignores all declines for leases in this prefix.

If disabled or not set, the DHCP server sets to UNAVAILABLE every address or delegated prefix requested in a DECLINE message if it is leased to the client.

The default value is false, so that DECLINE messages are processed normally.

**link** `nameref`(Link)

Associates an IPv6 prefix (subnet) with a link. Use this attr to group prefixes that are on a single link.

**local-cluster** `oid`

Identifies the local DHCP cluster or failover pair for this regional prefix.

**max-leases** `rangeint`(0-2000000) default = 65536

Sets the maximum number of non-reserved leases that the server will allow to exist on this prefix. When a new lease needs to be created, the server will only do so if the limit has not been exceeded. When the limit is exceeded, no new leases can be created and offered to clients.

This limit is not applied when existing leases are loaded from the lease state database during server start-up.

This attribute is also used when calculating the free-address level for address traps.

**name** `string` required,unique,immutable

Assigns a name to an IPv6 prefix (subnet).

**policy** `nameref`(Policy)

Refers to a shared policy to use when replying to clients.

**range** `prefix`

Specifies a subrange the server can use to configure prefixes for address assignment.

The prefix used depends on the value set for the dhcp-type at If not set, the prefix address is used. This may specify a local prefix than the prefix address to limit the range of addresses or prefixes available for assignment.

**reverse-zone-prefix-length** `rangeint`(0-124)

Specifies the prefix length of the reverse zone for ip6.arpa updates. You do not need to specify the full reverse zone, because you can synthesize it by using the ip6.arpa domain.

Use a multiple of 4 for the value, because ip6.arpa zones are on 4-bit boundaries. If not a multiple of 4, the value is rounded up to the next multiple of 4.

The maximum value is 124, because specifying 128 would create a zone name without any possible hostnames contained within.

A value of 0 means none of the bits are used for the zone name, hence ip6.arpa is used.

If you omit the value from the DNS update configuration, the server uses the value from the prefix or, as a last resort, the prefix length derived from the address value of the prefix.

**selection-tags** `string`

Associates selection tags with an IPv6 prefix (subnet).

**vpn-id** **int** default = 0, immutable

Identifies the VPN that contains an IPv6 prefix.

## prefix-policy

prefix-policy - Edits a DHCP policy embedded in a prefix

Note: dhcp-prefix-policy is a synonym for compatibility with earlier versions of Network Registrar.

### Synopsis

```
prefix-policy <name> delete
prefix-policy <name> set <attribute>=<value>
                    [<attribute>=<value> ...]
prefix-policy <name> get <attribute>
prefix-policy <name> disable <feature>
prefix-policy <name> enable <feature>
prefix-policy <name> show

prefix-policy <name> setV6Option <opt-name | id> <value>
prefix-policy <name> getV6Option <opt-name | id>
prefix-policy <name> unsetV6Option <opt-name | id>
prefix-policy <name> listV6Options

prefix-policy <name> setV6VendorOption <opt-name | id>
                                       <opt-set-name> <value>
prefix-policy <name> getV6VendorOption <opt-name | id>
                                       <opt-set-name>
prefix-policy <name> unsetV6VendorOption <opt-name | id>
                                       <opt-set-name>
prefix-policy <name> listV6VendorOptions
```

### Description

The prefix-policy command lets you configure a DHCP policy that is embedded in a DHCP prefix. An embedded policy is a collection of DHCP option values and settings associated with (and named by) another object -- in this case a prefix. A prefix-policy is created implicitly when you first reference it, and is deleted when the prefix is deleted.

You can set individual option values with the setV6Option command, unset option values with the unsetV6Option command, and view option values with the getV6Option and listV6Options commands. When you set an option value the DHCP server will replace any existing value or create a new one as needed for the given option name.

### Examples

### Status

### See Also

[policy](#), [client-policy](#), [client-class-policy](#), [dhcp-address-block-policy](#), [link-policy](#),

[policy](#), [scope-template-policy](#)

## Attributes

### **affinity-period** *time*

Associates a lease in the AVAILABLE state with the client last held the lease. If the client requests a lease during affinity period, it is granted the same lease; that is, renewals are prohibited, then it is explicitly not given a new lease. Because of the vast IPv6 address space and depending on the generation technique, it could be millions of years before an address ever needs reassignment to a different client, and there is no reason to hold on to this information for that long. To prohibit renewals enable either the `inhibit-all-renewals` or the `inhibit-renewals-at-reboot` attribute.

### **allow-client-a-record-update** *bool* default = disabled

Determines if a client is allowed to update A records. If the client sets the flags in the FQDN option to indicate it wants to do the A record update in the request, and if the value is TRUE, the server allows the client to do the A record update; otherwise, based on other server configurations, the server does the A record update.

### **allow-dual-zone-dns-update** *bool* default = disabled

Enables DHCP clients to perform DNS updates into two DNS zones. To support these clients, you can configure the DHCP server to allow the client to perform an update, but also to perform a reverse update on the client's behalf.

### **allow-lease-time-override** *bool* default = disabled

Gives the server control over the lease period. Although a client can request a specific lease time, the server need not honor the request if this attribute is set to false (the default). Even if set to true, clients can request only lease times shorter than those configured for the server.

### **allow-non-temporary-addresses** *bool* default = true

Determines whether DHCPv6 clients can request non-temporary (IA\_NA) addresses.

The default is to allow clients to request non-temporary addresses.

### **allow-rapid-commit** *bool* default = false

Determines whether DHCPv6 clients can use a Solicit with the Rapid Commit option to obtain configuration information with fewer messages. To permit this, make sure that a single DHCP server is servicing clients.

This attribute has special handling during the policy hierarchy processing when checking the Prefix policies (embedded or for the Prefixes on a Link. The Prefixes for the Link are processed in alphabetic (case blind) order. For each Prefix, the embedded and then named policy are checked. Only Prefixes the client has access (based on selection tags, etc.) are checked.

- If any of the prefix policies has this attribute set to FALSE, Rapid Commit is not allowed.
- If at least one has it set to TRUE, Rapid Commit is allowed.
- Otherwise, the remaining policies in the hierarchy are checked.

The default is not to allow clients to use Rapid Commit.

### **allow-temporary-addresses** *bool* default = true

Determines whether DHCPv6 clients can request temporary (floating) addresses.

The default is to allow clients to request temporary addresses.

**default-prefix-length** `rangeint(0-128)` default = 64

For delegation, specifies the default length of the delegation prefix if a router (client) does not explicitly request it.

The default length must always be less than or equal to the length of the prefix range.

**forward-dnsupdate** `nameref(DnsUpdateConfig)`

Specifies the name of the update configuration that determines which forward zones to include in updates.

**forward-zone-name** `dname`

Designates an optional forward zone for DNS updates.

**giaddr-as-server-id** `bool` default = false

Enables the DHCP server to set the server-id option on a DHCPDISCOVER and a DHCPACK to the giaddr of the incoming packet, instead of the IP address of the server (the default action).

This causes all unicast renews to be sent to the relay agent instead of directly to the DHCP server, and so renews arrive at the DHCP server with option-82 information appended to the packet. Some relay agents may not support this capability and, in complex configurations, the giaddr might not actually be the address to which the DHCP client can send a unicast packet. In the absence of the giaddr, the DHCP client cannot renew a lease, and must always perform a rebind operation (where the DHCP client broadcasts a request for a new lease) instead of sending a unicast to what it believes is the DHCP server.

**grace-period** `time` default = 5m

Defines the length of time between the expiration of a lease and the time it is made available for reassignment.

**inhibit-all-renews** `bool` default = false

Causes the server to reject all renewal requests, forcing clients to obtain a different address any time it contacts the DHCP server.

**inhibit-renews-at-reboot** `bool` default = false

Permits clients to renew their leases, but the server forces them to obtain new addresses each time they reboot.

**limitation-count** `int`

Specifies the maximum number of clients with the same limitation that are allowed to have currently active and valid leases.

**offer-timeout** `time` default = 2m

Instructs the server to wait a specified amount of time when it has offered a lease to a client, but the offer is not yet accepted. At the end of the specified time interval, the server makes the lease available again.

**packet-file-name** `string`

Identifies the boot-file to use in the boot process of a client. The server returns this file name in the 'file' field of its replies. The packet-file-name cannot be longer than 128 characters.

**packet-server-name** `string`

Identifies the host-name of the server to use in a client boot process. The server returns this file name in the 'sname' field of its replies. The packet-server-name field cannot be longer than 64 characters.

**packet-siaddr** `ipaddr`

Identifies the IP address of the next server in the client process. For example, this might be the address of a TFTP used by BOOTP clients. The server returns this address in 'siaddr' field of its replies.

### **permanent-leases** `bool` default = disabled

Indicates whether leases using this policy are permanently granted to requesting clients. If leases are permanently granted, the server ignores the configured `dhcp-lease-time` option value and uses the maximum lease time.

### **preferred-lifetime** `time` default = 1w

Assigns the default and maximum preferred lifetime for leases on DHCPv6 client interfaces. Expressed in seconds and relative to the time the server sent the packet, this attribute sets the length of time that the address is preferred; that is, it is unrestricted. When the preferred lifetime expires, the address becomes deprecated and its use is restricted.

### **reconfigure** `enumint`(allow=1, disallow=2, require=3) default = allow

Controls DHCPv6 client reconfiguration support:

- |   |          |                                                                                                                                                               |
|---|----------|---------------------------------------------------------------------------------------------------------------------------------------------------------------|
| 1 | allow    | Allows clients to request reconfiguration support and the server will honor the request (default).                                                            |
| 2 | disallow | Allows clients to request reconfiguration support but the server will not honor the clients' request.                                                         |
| 3 | require  | Requires clients to request reconfiguration support and the server drops client Solicit and Request messages that do not include a Reconfigure-Accept option. |

This attribute has special handling during the policy hierarchy processing when checking the Prefix policies (embedded or for the Prefixes on a Link. The Prefixes for the Link are processed in alphabetic (case blind) order. For each Prefix embedded and then named policy are checked. Only Prefixes the client has access (based on selection tags, etc.) are as follows:

- If any of the prefix policies has this attribute set to disallow or require, that setting is used.
- Otherwise, if at least one has it set to allow, Reconfigure is allowed.
- If no prefix policies have this attribute set, the reconfigure policies in the hierarchy are checked.

### **reconfigure-via-relay** `bool` default = false

Controls whether the server should prefer unicasting or relaying DHCPv6 Reconfigure messages.

If false (the default), the server prefers to unicast Reconfigure messages if the client has one or more valid statefully assigned addresses.

If true, the server prefers to send Reconfigure messages via the relay agent unless no relay agent information is available.

Note: When you use this attribute, consider that:

- In networks where the DHCPv6 server cannot communicate directly with its client devices?for example, where firewalls are in place?set this value to true.
- The DHCPv6 server does not use embedded and named policies configured on a client when it evaluates this attribute.

- The relay agent cannot be used if the Relay-Forw m came from a link-local address.

### **reverse-dnsupdate [nameref](#)(DnsUpdateConfig)**

Specifies the name of the update configuration that determines which reverse zones to include in a DNS update.

### **server-lease-time [time](#)**

Tells the server how long a lease is valid. For more frequent communication with a client, you might have the server consider leases as leased for a longer period than the client considers. This also provides more lease-time stability. This value is used unless it is longer than the lease time in the dhcp-lease-time found through the normal traversal of policies.

### **split-lease-times [bool](#) default = disabled**

Specifies a value that the DHCP server might use internally to affect lease times.

If enabled, the DHCP server still offers clients lease times that reflect the configured lease-time option from the appropriate policy; but the server bases its decisions regarding expiration on the 'server-lease-time' value.

### **unavailable-timeout [time](#) default = 24h**

Permits the server to make a lease unavailable for the time specified and then to return the lease to available state. If there is no value configured in the system\_default\_policy, then the default is 86400 seconds (or 24 hours).

### **use-client-id-for-reservations [bool](#) default = off**

Controls how the server database checks for reserved IP addresses.

By default, the server uses the MAC address of the DHCP client as the key for its database lookup. If this attribute is set to on (enabled), then the server does the check for reserved addresses using the DHCP client-id, which the client usually provides. In cases where the DHCP client does not supply the client-id, the server synthesizes it, and uses that value.

### **v4-bootp-reply-options [optionid4](#)**

Lists the options the server returns to all BOOTP clients.

### **v4-reply-options [optionid4](#)**

Lists the options the server returns to all DHCPv4 clients, regardless of whether the client specifically asks for the option data.

### **v6-reply-options [optionid6](#)**

Lists the options that should be returned in any replies to DHCPv6 clients.

This attribute has special handling during the policy hierarchy processing when checking the Prefix policies (embedded or not) for the Prefixes on a Link. The Prefixes for the Link are processed in alphabetic (case blind) order. For each Prefix, the embedded and then named policy are checked. Only Prefixes that the client has access to (based on selection tags, etc.) are returned.

### **valid-lifetime [time](#) default = 2w**

Assigns the default and maximum valid lifetime for leases on DHCPv6 client interfaces. Expressed in seconds and relative to the time the server sent the packet, this attribute sets the length of time that an address remains valid. When the period of time expires, the address becomes invalid and unusable. The valid lifetime must be greater than or equal to the preferred lifetime.

# prefix-template

prefix-template - Configures a prefix template.

## Synopsis

```
prefix-template <name> create [<attribute>=<value> ...]
prefix-template <name> delete
prefix-template <name> set <attribute>=<value> [<attribute>=<value> ...]
prefix-template <name> get <attribute>
prefix-template <name> unset <attribute>

prefix-template <name> disable <attribute>
prefix-template <name> enable <attribute>
prefix-template <name> show

prefix-template <name> create clone=<clone-name>
prefix-template <name> apply-to <<b>all | <prefix1>[,...]>
```

## Description

The prefix-template command lets you configure a template to use when creating prefixes.

## Examples

## Status

## See Also

[prefix](#)

## Attributes

**allocation-algorithms flags**(client-request=1, reservation=2, extension=3 identifier=4, random=5, best-fit=6) default = reservation,extension,random

Controls the algorithms used by the server to select a or prefix to lease to a client. The available algorithm

client-request

This setting (off by default) controls whether the uses a client requested lease.

reservation

This setting (on by default) controls whether the uses an available reservation for the client.

extension

This setting (on by default) controls whether the extensions attached at the generate-lease extensio to generate an address or prefix for the client.

interface-identifier

This setting (off by default) controls whether the the interface-identifier from the client's (link-1 address to generate an address. It is ignored for addresses and prefix delegation.

random

This setting (on by default) controls whether the generates an address using an RFC 3041-like algori is ignored for prefix delegation.

best-fit

This setting (on by default) controls whether the delegate the first, best-fit available prefix. It for addresses.

When the server needs an address to assign to a client processes the flags in the following order: client-req reservation, extension, interface-identifier, and rand Processing stops when a usable address is produced.

When the server needs to delegate a prefix to a client processes the flags in the following order: client-req reservation, extension, best-fit. Processing stops whe usable prefix is produced.

**deactivated** `bool` default = disabled

Controls whether a prefix extends leases to clients. A prefix does not extend leases to any clients. It treat in its ranges as if they were individually deactivated Default, false (active).

**description** `string`

Describes the prefix template.

**dhcp-type** `enumint`(stateless=0, dhcp=1, prefix-delegation=2, infrastructure default = dhcp

Defines how DHCP manages address assignment within a p

stateless	Prefix is used only for stateless option configuration.
dhcp	Prefix is used for stateful address assignment.
prefix-delegation infrastructure	Prefix is used for prefix-delegati Prefix is used to map a client add a link, and does not have an addre
parent	Prefix is not used by DHCP. It is a container object, to group child

**expiration-time** `date`

Sets the time and date on which a prefix expires. Afte and time, the server neither grants new leases nor ren leases from this prefix.

Once the expiration-time has passed, the prefix is no (though old leases and leases with grace or affinity p continue to exist until those periods elapse).

Enter this as a date in the format "[weekday] mon day hh:mm[:ss] year". For example, "Dec 31 23:59 2006".

**free-address-config** `nameref`(AddrTrapConfig)

Identifies which trap captures unexpected free address on this prefix.

If this attribute is not configured, the server looks the free-address-config attribute on the parent Link c If that attribute is not configured, the server looks the v6-default-free-address-config on the DHCPServer o

**ignore-declines** `bool` default = false

Controls whether the DHCP server responds to a DHCPv6 message that refers to an IPv6 address or a delegated from this prefix.

If enabled, the DHCP server ignores all declines for 1 in this prefix.

If disabled or not set, the DHCP server sets to UNAVAI every address or delegated prefix requested in a DECLI message if it is leased to the client.

The default value is false, so that DECLINE messages a

processed normally.

**max-leases** **rangeint**(0-2000000) default = 65536

Sets the maximum number of non-reserved leases that the server will allow to exist on the prefix. When a new lease needs to be created, the server will only do so if the limit has been exceeded. When the limit is exceeded, no new leases are created and offered to clients.

**name** **string** required,unique,immutable

Assigns a name to this prefix template.

**options-expr** **expr**

Defines an expression that evaluates to the list of embedded policy options to be created.

**policy** **nameref**(Policy)

Refers to a shared policy to use when replying to clients.

**prefix-description-expr** **expr**

Defines an AT\_STRING expression to apply to the description on the prefix object created when using the template.

**prefix-name-expr** **expr**

Defines an expression that evaluates to an AT\_STRING value to use for the name of the prefix object created when using the template.

**range-expr** **expr**

Defines an expression that evaluates to an AT\_PREFIX value for the prefix range to be created.

**reverse-zone-prefix-length** **rangeint**(0-124)

Identifies the prefix length of the reverse zone for IPv6 updates. The server forms the zone name using this value if configured; otherwise the prefix length is determined by the reverse zone. This value must be a multiple of 4 as IPv6 addresses are on 4 bit (nibble) boundaries. If not a multiple of 4, it is rounded up to the next higher multiple of 4.

**selection-tags** **string**

Associates selection tags with an IPv6 prefix.

## policy

policy - Specifies DHCP policy information

### Synopsis

```
policy <name> create [<attribute>=<value>]
policy <name> create clone=<clone-name>
policy <name> delete
policy list
policy listnames
policy <name> set =<value> [<attribute>=<value> ...]
policy <name> get <attribute>
policy <name> unset <attribute>

policy <name> disable <attribute>
policy <name> enable <attribute>
policy <name> show

policy <name> setLeaseTime <time-val>
policy <name> getLeaseTime
```

```

policy <name> setOption <opt-name | id> <value>
policy <name> getOption <opt-name | id>
policy <name> unsetOption <opt-name | id>
policy <name> listOptions

policy <name> setV6Option <opt-name | id> <value>
policy <name> getV6Option <opt-name | id>
policy <name> unsetV6Option <opt-name | id>
policy <name> listV6Options

policy <name> setVendorOption <opt-name | id> <opt-set-name> <value>
policy <name> getVendorOption <opt-name | id> <opt-set-name>
policy <name> unsetVendorOption <opt-name | id> <opt-set-name>
policy <name> listVendorOptions

policy <name> setV6VendorOption <opt-name | id> <opt-set-name> <value>
policy <name> getV6VendorOption <opt-name | id> <opt-set-name>
policy <name> unsetV6VendorOption <opt-name | id> <opt-set-name>
policy <name> listV6VendorOptions

```

## Description

The `policy` command manages DHCP policy configurations. A policy is a collection of DHCP option values to associate with a range of addresses in a scope, or with a specific client or client-class configuration. Network Registrar considers policy reply options in a hierarchy of options. For details on these reply options, see the User Guide for Cisco Network Registrar.

The `policy` command by itself is for a named policy. You can also manage the embedded policies for `dhcp-address-block`, `client`, `client-class` and `scope` objects through the `dhcp-address-block-policy`, `client-policy`, `client-class-policy`, and `scope-policy` commands, respectively.

For embedded policies, `name` identifies the object that contains the embedded policy. For example, an `attribute-setting` command for a scope policy would be `"scope-policy scope-name set <attribute>"`--using the name of the scope for the `name` value.

The default policy is a special named policy that includes default settings. You can manage the default policy just like all the other named ones.

```

policy <name> setLeaseTime <time-val>
policy <name> getLeaseTime

```

Use the `setLeaseTime` command to set the values of lease times and the `getLeaseTime` command to display the value of a lease time.

```

policy <name> setV6Option <opt-name | id> <value>
policy <name> getV6Option <opt-name | id>
policy <name> unsetV6Option <opt-name | id>
policy <name> listV6Options

```

To manage the DHCPv6 options on the policy, use the commands: `setV6Option`, `getV6Option`, `unsetV6Option`, `listV6Options`.

```
policy <name> setVendorOption <opt-name | id> <opt-set-name> <value>
policy <name> getVendorOption <opt-name | id> <opt-set-name>
policy <name> unsetVendorOption <opt-name | id> <opt-set-name>
policy <name> listVendorOptions
```

The `setVendorOption` and `getVendorOption` commands are used to set and get vendor-specific option data on the policy. These commands require an option name and the name of a vendor-specific option definition set.

The `unsetVendorOption` command removes the data for the specific vendor option.

The `listVendorOptions` command displays all vendor-option data that is set in the policy. The listing includes the name of the option-definition set that was used to define the data.

You can manage the DHCPv6 vendor options on the policy using the commands: `setV6VendorOption`, `getV6VendorOption`, `unsetV6VendorOption`, `listV6VendorOptions`.

NOTE: The vendor option commands use a different syntax when connected to releases prior to 6.2. See the documentation for the specific release.

## Examples

```
nrcmd> policy default setOption dhcp-lease-time 608400
nrcmd> policy default listOptions
      (51)dhcp-lease-time: 604800
nrcmd> policy default set grace-period=3d
```

## Status

## See Also

[option-set](#), [option](#)

## Attributes

### **affinity-period time**

Associates a lease in the AVAILABLE state with the client that last held the lease. If the client requests a lease during affinity period, it is granted the same lease; that is, unless renewals are prohibited, then it is explicitly not given the lease. Because of the vast IPv6 address space and depending on the generation technique, it could be millions of years before the address ever needs reassignment to a different client, and there is no reason to hold on to this information for that long. To prohibit renewals enable either the `inhibit-all-renews` or the `inhibit-renews-at-reboot` attribute.

### **allow-client-a-record-update bool default = disabled**

Determines if a client is allowed to update A records. If the client sets the flags in the FQDN option to indicate it wants to do the A record update in the request, and if the value is TRUE, the server allows the client to do the A record update; otherwise, based on other server configurations, the

does the A record update.

**allow-dual-zone-dns-update** **bool** default = disabled

Enables DHCP clients to perform DNS updates into two DNS zones. To support these clients, you can configure the DHCP server to allow the client to perform an update, but also to perform update on the client's behalf.

**allow-lease-time-override** **bool** default = disabled

Gives the server control over the lease period. Although a client can request a specific lease time, the server need not honor request if this attribute is set to false (the default). Even if set to true, clients can request only lease times that are shorter than those configured for the server.

**allow-non-temporary-addresses** **bool** default = true

Determines whether DHCPv6 clients can request non-temporary (IA\_NA) addresses.

The default is to allow clients to request non-temporary addresses.

**allow-rapid-commit** **bool** default = false

Determines whether DHCPv6 clients can use a Solicit with the Rapid Commit option to obtain configuration information with fewer messages. To permit this, make sure that a single DHCP server is servicing clients.

This attribute has special handling during the policy hierarchy processing when checking the Prefix policies (embedded or not) for the Prefixes on a Link. The Prefixes for the Link are processed in alphabetic (case blind) order. For each Prefix, the embedded and then named policy are checked. Only Prefixes that the client has access (based on selection tags, etc.) are checked.

- If any of the prefix policies has this attribute set to FALSE, Rapid Commit is not allowed.
- If at least one has it set to TRUE, Rapid Commit is allowed.
- Otherwise, the remaining policies in the hierarchy are checked.

The default is not to allow clients to use Rapid Commit.

**allow-temporary-addresses** **bool** default = true

Determines whether DHCPv6 clients can request temporary (IA\_TA) addresses.

The default is to allow clients to request temporary addresses.

**default-prefix-length** **rangeint(0-128)** default = 64

For delegation, specifies the default length of the delegation if a router (client) does not explicitly request it. The default length must always be less than or equal to the length of the prefix range.

**forward-dnsupdate** **nameref(DnsUpdateConfig)**

Specifies the name of the update configuration that determines which forward zones to include in updates.

**forward-zone-name** **dname**

Designates an optional forward zone for DNS updates.

**giaddr-as-server-id** **bool** default = false

Enables the DHCP server to set the server-id option on a DHCPACK and a DHCPACK to the giaddr of the incoming packet, instead of the IP address of the server (the default action).

This causes all unicast renews to be sent to the relay agent directly to the DHCP server, and so renews arrive at the server with option-82 information appended to the packet. Some relay agents may not support this capability and, in such cases, renews are sent to the DHCP server via the relay agent.

complex configurations, the giaddr might not actually be an address to which the DHCP client can send a unicast packet. In these cases, the DHCP client cannot renew a lease, and must always perform a rebinding operation (where the DHCP client broadcasts a request for sending a unicast to what it believes is the DHCP server).

**grace-period** **time** default = 5m

Defines the length of time between the expiration of a lease and the time it is made available for reassignment.

**inhibit-all-renews** **bool** default = false

Causes the server to reject all renewal requests, forcing clients to obtain a different address any time it contacts the DHCP server.

**inhibit-renews-at-reboot** **bool** default = false

Permits clients to renew their leases, but the server forces them to obtain new addresses each time they reboot.

**limitation-count** **int**

Specifies the maximum number of clients with the same limitation that are allowed to have currently active and valid leases.

**offer-timeout** **time** default = 2m

Instructs the server to wait a specified amount of time when it has offered a lease to a client, but the offer is not yet accepted. At the end of the specified time interval, the server makes the lease available again.

**packet-file-name** **string**

Identifies the boot-file to use in the boot process of a client. The server returns this file name in the 'file' field of its replies. The packet-file-name cannot be longer than 128 characters.

**packet-server-name** **string**

Identifies the host-name of the server to use in a client's boot process. The server returns this file name in the 'sname' field of its replies. The packet-server-name field cannot be longer than 64 characters.

**packet-siaddr** **ipaddr**

Identifies the IP address of the next server in the client's boot process. For example, this might be the address of a TFTP server used by BOOTP clients. The server returns this address in the 'siaddr' field of its replies.

**permanent-leases** **bool** default = disabled

Indicates whether leases using this policy are permanently granted to requesting clients. If leases are permanently granted, the server ignores the configured dhcp-lease-time option value and uses the maximum lease time.

**preferred-lifetime** **time** default = 1w

Assigns the default and maximum preferred lifetime for leases on DHCPv6 client interfaces. Expressed in seconds and relative to the time the server sent the packet, this attribute sets the length of time that the address is preferred; that is, its use is unrestricted. When the preferred lifetime expires, the address becomes deprecated and its use is restricted.

**reconfigure** **enumint**(allow=1, disallow=2, require=3) default = allow

Controls DHCPv6 client reconfiguration support:

1	allow	Allows clients to request reconfiguration support and the server will honor the request (default).
2	disallow	Allows clients to request reconfiguration support but the server will not honor the request.

- support but the server will not honor the clients' request.
- 3 require Requires clients to request reconfiguration support and the server drops client Solicit and Request messages that do not include a Reconfigure-Accept option.

This attribute has special handling during the policy hierarchy processing when checking the Prefix policies (embedded or r for the Prefixes on a Link. The Prefixes for the Link are processed in alphabetic (case blind) order. For each Prefix embedded and then named policy are checked. Only Prefixes that the client has access (based on selection tags, etc.) are c as follows:

- If any of the prefix policies has this attribute set to disallow or require, that setting is used.
- Otherwise, if at least one has it set to allow, Reconfigure is allowed.
- If no prefix policies have this attribute set, the remaining policies in the hierarchy are checked.

### **reconfigure-via-relay** **bool** default = false

Controls whether the server should prefer unicasting or relaying DHCPv6 Reconfigure messages.

If false (the default), the server prefers to unicast Reconfigure messages if the client has one or more valid statefully assigned addresses.

If true, the server prefers to send Reconfigure messages via the relay agent unless no relay agent information is available.

Note: When you use this attribute, consider that:

- In networks where the DHCPv6 server cannot communicate directly with its client devices?for example, where firewalls are in place?set this value to true.
- The DHCPv6 server does not use embedded and named policies configured on a client when it evaluates this attribute.
- The relay agent cannot be used if the Relay-Forw message came from a link-local address.

### **reverse-dnsupdate** **nameref**(DnsUpdateConfig)

Specifies the name of the update configuration that determines which reverse zones to include in a DNS update.

### **server-lease-time** **time**

Tells the server how long a lease is valid. For more frequent communication with a client, you might have the server consider leases as leased for a longer period than the client considers. This also provides more lease-time stability. This value is used unless it is longer than the lease time in the dhcp-lease-time found through the normal traversal of policies.

### **split-lease-times** **bool** default = disabled

Specifies a value that the DHCP server might use internally to affect lease times.

If enabled, the DHCP server still offers clients lease times that reflect the configured lease-time option from the appropriate policy; but the server bases its decisions regarding expiration on the 'server-lease-time' value.

### **unavailable-timeout** **time** default = 24h

Permits the server to make a lease unavailable for the time specified and then to return the lease to available state. If there is

configured in the `system_default_policy`, then the default is 86400 seconds (or 24 hours).

#### **use-client-id-for-reservations** `bool` default = off

Controls how the server database checks for reserved IP addresses.

By default, the server uses the MAC address of the DHCP client key for its database lookup. If this attribute is set to `on` (enabled), then the server does the check for reserved addresses using the DHCP client-id, which the client usually supplies. In cases where the DHCP client does not supply the client-id, the server synthesizes it, and uses that value.

#### **v4-bootp-reply-options** `optionid4`

Lists the options the server returns to all BOOTP clients.

#### **v4-reply-options** `optionid4`

Lists the options the server returns to all DHCPv4 clients, or not the client specifically asks for the option data.

#### **v6-reply-options** `optionid6`

Lists the options that should be returned in any replies to DHCPv6 clients.

This attribute has special handling during the policy hierarchy processing when checking the Prefix policies (embedded or referenced) for the Prefixes on a Link. The Prefixes for the Link are processed in alphabetic (case blind) order. For each Prefix, the embedded and then named policy are checked. Only Prefixes that the client has access (based on selection tags, etc.) are checked.

#### **valid-lifetime** `time` default = 2w

Assigns the default and maximum valid lifetime for leases to DHCPv6 client interfaces. Expressed in seconds and relative to the time the server sent the packet, this attribute sets the length of time that an address remains valid. When this period of time expires, the address becomes invalid and unusable. The valid lifetime must be greater than or equal to the preferred lifetime.

## region

`region` - Configures a named geographic region

### Synopsis

```
region <tag> create <name> [<attribute>=<value>]
region <tag> delete
region list
region listnames
region <tag> show
region <tag> set <attribute>=<value> [<attribute>=<value> ...]
region <tag> get <attribute>
region <tag> unset <attribute>

region <tag> enable <attribute>
region <tag> disable <attribute>
```

### Description

The `region` command configures objects representing a geographic region containing other objects, such as address blocks, subnets and zones.

## Examples

## Status

## See Also

## Attributes

**contact** [string](#)

The contact information for this region.

**name** [string](#) required

The full name or printable name for this region.

**tag** [string](#) required,unique

The unique tag name for this region. Typically a short name referring to this region.

## remote-dns

remote-dns - Specifies information about remote DNS servers for IXFR

### Synopsis

```
remote-dns list
remote-dns listnames
remote-dns <addr>[/<mask>] create [<attribute>=<value>...]
remote-dns <name> delete
remote-dns <name> disable <attribute>
remote-dns <name> enable <attribute>
```

### Description

The remote-dns command lets you control the behavior of the DNS server when it is querying other DNS servers. Currently, the only configuration properties are controls on incremental transfer (ixfr) and the sending of multiple records per TCP "packet" (multirec).

The names of the remote-dns configurations provide the address and netmask to use when matching addresses to configurations. In the case of multiple possible matches, the DNS server uses the address with the largest bitmask.

### Examples

To enable anyone on the 192.168.0.0 network to perform incremental transfers except 192.168.1.1:

```
nrcmd> remote-dns create 192.168.0.0/16 ixfr=true
nrcmd> remote-dns create 192.168.1.1/32 ixfr=false
```

### Status

### See Also

## Attributes

### **ixfr** **bool** default = disabled, required

Indicates whether or not a foreign server supports incremental transfer and should be queried for incremental (IXFR) be (AXFR) when asking for zone transfers. Although unwittir this to true is generally harmless, doing so may result additional transactions to accomplish a zone transfer.

### **key** **dname**

Specifies which key to use when communicating with the r server.

### **multirec** **bool** default = disabled, required

Indicates whether or not a foreign server can be sent zone transfers (AXFR) with multiple records in one TCP "packe DNS servers will crash when receiving such transfers, de being allowed by the protocol specification.

## report

`report` - Creates a summary report of address usage

### Synopsis

```
report [column-separator=<string>]
        [dhcp-only]
        [dhcpv4]
        [dhcpv6]
        [file=<output file>]
        [vpn=<vpn-name>]
```

### Description

The `report` command provides address utilization reporting for all subnets containing addresses that are handled by the DHCP server. The DHCP server must be running to obtain the report or a "Connection Failure" error message will be returned.

The `report` command provides v4 and v6 utilization. The option 'dhcp-only' will be kept as a synonym for 'dhcpv4', to provide backwards compatibility.

The output for the `report dhcpv4` command is a table consisting of column-aligned data for each defined scope, subnet, and CCM address block. Subtotals are listed for subnets and address blocks that contain multiple scopes or subnets. A grand total row summarizes the data from all the subnets.

Each row lists the following information:

- o Subnet/Mask
- o Scope Name
- o %Free - percentage of dynamic addresses available to be leased
- o Total Dynamic - total number of addresses configured, excluding reservations
- o Total Reserved - total reserved addresses
- o Leased - addresses actively leased by clients
- o Avail - addresses available to be leased
- o Other Avail - addresses set aside to be leased by this server's

- failover partner
- o Pending Avail - leases not available to be leased because the server is in the communications-interrupted failover state.
- o In Transition - leases offered to clients or waiting for the configured grace period before again becoming available
- o Reserved Active - reserved addresses actively leased by clients
- o Unavail - addresses marked unavailable
- o Active Deactivated - addresses that have been administratively deactivated

The output for the report `dhcpv6` command is a table consisting of column-aligned data for each defined prefix. If requested, totals will be computed for each parent prefix defined in the address tree. Separate total lines will be shown for child prefixes classified for prefix delegation or dhcp allocation. No counters apply for stateless configuration or infrastructure entries. The only attribute that may be reported is whether the prefix has been deactivated. Total lines will be skipped if the prefix-count for a given type is 0.

Each row lists the following information:

- o Prefix/Mask
- o Prefix Name
- o Total Reserved - total number of reserved leases configured in this prefix
- o Leased - addresses actively leased by clients
- o Revoked - dynamic leases (leases that are not reserved) that are no longer usable by the client, but that may still be in use
- o In Transition - leases offered to clients or waiting for the configured grace period before again becoming available
- o Reserved Active - reserved addresses actively leased by clients
- o Reserved Inactive - reserved addresses not used by their assigned DHCP clients
- o Reserved Unavail - reserved addresses marked unavailable
- o Unavail - addresses marked unavailable
- o Active Dynamic - total number of dynamic leases actively in use
- o Active Deactivated - number of dynamic and reserved leases currently leased by DHCP clients, but have also been administratively deactivated
- o Deactivated - addresses that have been administratively deactivated

## Examples

```
report
report file=myreport.txt
```

## Status

## See Also

[export](#) addresses, [lease-notification](#), [session](#) current-vpn

## Report Keywords

### column-separator

Specifies the character string you want used between the columns in the report. The default is a single space. If you specify more than one space, you must use a backslash (\) to allow the extra spaces, and if you enter the spaces on the command line, use quotation marks.

### dhcp-only

Provided for command-syntax compatibility with prior versions, specifies a summary of the DHCP server information. This is the only option available for the report command, and is no longer required to run the command.

### dhcpv4

Displays ipv4 utilization.

### dhcpv6

Display ipv6 utilization.

### file

Specifies the filename to which the report command writes the output. If you do not specify a filename, the report command writes to 'standard out'.

### vpn

The VPN address space from which to select scopes to examine when executing this command. If no vpn-name is specified, then the session's current-vpn is used. If the reserved vpn-name "global" is used, then the global (or unnamed) VPN address space is used. The reserved vpn-name "all" is not allowed for this command, because the report command has no mechanism to distinguish identical IP addresses in different VPNs.

## reservation

reservation - Configures DHCP reservations

### Synopsis

```
reservation [<vpn-name>/]<ipaddr> create (<macaddr>|<lookup-key>)  
                [-mac|--blob|-string] [attribute=<value>...]  
reservation [<vpn-name>/]<ipaddr> delete  
reservation [<vpn-name>/]<ipaddr> get <attribute>  
reservation [<vpn-name>/]<ipaddr> show  
reservation list [[<vpn-name>/]<ipaddr>|-mac|-key]
```

### Description

The reservation command lets you manipulate Network Registrar's global list of reservations. Changes to reservations also modify the reservations listed with each scope (and vice versa). Thus, these commands are an alternative to the scope's addReservation, removeReservation, and listReservation commands.

A matching scope must exist for each reservation in the global list. When you create or delete reservations, they are added or removed from the appropriate scope. When multiple scopes exist

for the same subnet, you must specify the `include-tags` attribute to match the reservation to the appropriate scope. If no matching scope is found, the edit is rejected as invalid.

The `reservation list` command displays the reservations in address order unless `-mac` or `-key` is specified to change the sort order.

## Examples

## Status

## See Also

## Attributes

### **client-class** [nameref](#)(ClientClass)

Identifies the client-class to use when selecting from a scopes that might contain a specified reservation.

### **description** [string](#)

Describes the device that this reservation object repres

### **device-name** [string](#)

Displays the name of the device represented by this reservation object.

### **include-tags** [string](#)

Displays the selection criteria for this reservation.

### **ip6address** [ip6net](#)

Specifies the IPv6 address for the reservation.

### **ipaddr** [ipaddr](#)

Displays an IP address within the network that the scope and that contains the reservation.

### **lookup-key** [blob](#)

Displays the sequence of bytes which is the key for this object. If this attribute appears in the reservation object value overrides any value in the `mac` attribute. The type object is found in the attribute `lookup-key-type`.

### **mac** [macaddr](#)

Displays the MAC address of the client for whom the specified IP address is to be reserved.

### **relay-info** [option](#)

Sets the contents of DHCP option-82 (`relay-agent-info`) in a synthesized lease. For `leasequery` to work with reservations for static addresses in a cable environment, the `relay-agent-info` information needs to be configured with the reservation. The attribute should contain all the contents of the `relay-agent-info` (option-82) to be provided in a synthesized `leasequery` response for this reservation.

### **vpn-id** [int](#) default = 0

Displays the identifier of the VPN that contains this re

# role

role - Configures a role

## Synopsis

```
role <name> create <base-role> [<attribute>=<value>]
role <name> delete
role list
role listnames
role <name> show
role <name> set =<value> [<attribute>=<value> ...]
role <name> get <attribute>

role <name> enable <attribute>
role <name> disable <attribute>
```

## Description

The role command configures the specified role. A role describes the operations that an administrator can perform and any data constraints that should be applied. A role must be assigned to an administrator group to be associated with an administrator.

The constraint property should be manipulated using the web UI.

## Examples

## Status

## See Also

[group](#), [admin](#)

## Attributes

### **all-sub-roles** [bool](#) default = true

Controls whether to ignore the sub-role attribute for this role. If this attribute is unset, or if it is set to true, then the role ignores the value of the sub-roles attribute and this subrole is authorized for all sub-roles. If this attribute is false, the sub-roles attribute provides the list of subroles for which this role instance is authorized. If the unconstrained attribute is set to true, then the values of this attribute and the sub-roles attribute are ignored, and the sub-role authorization for the role is for all sub-roles.

### **constraints** [obj\(0\)](#)

The list of constraints for the role.

### **groups** [string](#)

Lists the groups with which this role is associated. Any member of a listed group can perform the operations that the role allows.

### **name** [string](#) required,unique

Identifies the name of this role.

### **read-only** [bool](#) default = false

Indicates that all constraints associated with this role are limited to read-only access.

### **role** [string](#)

Specifies the base role for this object. The base role defines operations, such as modifying a zone, that are allowed and the further constraints on these operations. For example, a constrained role could limit the list of zones to a specified list of Owners.

### **sub-roles** [string](#)

Lists subroles associated with this role instance. If the all-sub-roles attribute is unset, or if it is set to true, then this attribute is ignored. If the all-sub-roles attribute is set to false, then this attribute specifies the list of subroles for this role instance, and an administrator associated with this role has authorization limited to the specified subroles. If the administrator has multiple roles in which the role attribute is the same, then subrole authorization for that role should be taken to be the union of all the set subroles from the individual role instances; and, if any of the role instances has the all-sub-roles attribute set to true, then subrole authorization for that role is for all subroles. Also, if any role instance for a matching role has the unconstrained attribute set to true, then subrole authorization for that role is for all subroles.

### **unconstrained** [bool](#) default = false

Indicates that this role has no other constraints beyond the list of operations it can perform.

## **router**

router - Configures a router

### **Synopsis**

```
router <name> create <address> [<attribute>=<value> ...]
router <name> delete
router list
router listnames
router <name> show
router <name> get <attribute>
router <name> set => [<attribute>= ...]
router <name> unset <attribute>
```

### **Description**

The router command lets you configure a router.

Be aware that while the type attribute is optional, it is needed to synchronize router-interface changes to the router.

### **Examples**

### **Status**

### **See Also**

[router-type](#), [router-interface](#)

### **Attributes**

**address** [ipaddr](#) required,unique

Sets the management interface address for this router.

**description** [string](#)

Describes this router.

**device-timeout** [int](#) default = 60

Indicates the timeout in seconds. The maximum timeout RIC server's communication handler will wait for data from the router.

**enable** [clrtxt](#) transient

Sets the enable password, in clear-text form. Avoid this field in normal use and never send the password over insecure links. Use this field to provide the clear-text password to the RIC server which intentionally does not have access to the secret storage module.

**enable-secret** [secret](#)

Identifies the secret containing the clear-text password for enabling super-user access to the router.

**interfaces** [obj](#)(CCMRouterInterface) transient

Lists objects describing the interfaces associated with this router. This is used when the RIC server is returning information about a router and its interfaces, or when the CCM server is returning a list of routers and interfaces for the UI to display a tree of routers and interfaces.

**login-temp-obj** [obj](#)(CCMRouterLoginTemplate) transient

Specifies the actual login template object. This attribute is used so that the CCM server can provide the login template to the stateless RIC server.

**login-template** [nameref](#)(CCMRouterLoginTemplate)

Names an optional login template that to use to further customize the RIC server's login and enable interaction sessions.

**name** [string](#) required,unique,immutable

Names this router.

**owner** [oid](#)

Controls the owner of this object. This owner field is used to group similarly owned routers and can be used to limit administrative access.

**password** [clrtxt](#) transient

Determines the password that authenticates the username. Avoid this field in normal use and never send the password over insecure links. Use this field to provide the clear-text password to the RIC server which intentionally does not have access to the secret storage module.

**password-secret** [secret](#)

Identifies the secret containing the clear-text password for authenticating the username.

**region** [oid](#)

Controls the region associated with this object. This region field is used to group similarly located routers and can be used to limit administrative access.

**type** [nameref](#)(CCMRouterType)

Specifies the type of the router. This is needed so that the RIC server can use the correct implementation of the

router-specific interface.

**use-ssh** [enumint](#)(disabled=0, optional=1, required=2) default = 1

Specifies whether the RIC server should (or must) use SSH when communicating with the router.

**username** [string](#)

Identifies the username to log in to this router.

**virtual-router** [bool](#) default = false, transient

Indicates that the Router object is not managed by the RIC server. This feature may be used when there is no network connectivity to the router, or when synchronization with the network configuration is not desired. A virtual router configuration is stored in the CCM database, but no attempt is made to push the configuration to the router, or to synchronize any configuration changes from the router.

## router-interface

router-interface - Configures an interface on a router

### Synopsis

```
router-interface <router> <name> create [<attribute>=...]
```

```
router-interface <router> <name> delete
```

```
router-interface <router> list
```

```
router-interface <router> listnames
```

```
router-interface <router> <name> show
```

```
router-interface <router> <name> get <attribute>
```

```
router-interface <router> <name> set <attribute>=<value> [<attribute>=<value>
```

```
router-interface <router> <name> unset <attribute>
```

### Description

The router-interface command lets you configure an interface on a router.

### Examples

### Status

### See Also

[router](#), router-interface-type

### Attributes

**bundle-id** [int](#)

The id of the bundle for grouping bundled interfaces.

**cable-dhcp-giaddr** [enumint](#)(policy=2, primary=1, not-present=0) default

The setting for giaddr selection in cable interfaces.

**cable-helper** [ipaddr](#)

The list of ip addresses stored as the cable-helper addresses on the interface.

**description** [string](#)

The description of this interface.

**ip-helper** [ipaddr](#)

The list of ip addresses stored as the ip-helper v the interface.

**is-master** [bool](#) default = false

Indication that this is the master interface in a interfaces.

**is-virtual** [bool](#) default = false

Indication if this interface is a virtual sub-inte

**mac-address** [macaddr](#) immutable

The MAC address of this interface.

**name** [string](#) required,immutable

The name of the router interface.

**parent** [oid](#)(CCMRouterInterface)

The parent interface if we have sub-interfaces or interfaces configured on the router.

**primary-subnet** [net](#)

The primary subnet (and interface address) for thi interface.

**router** [oid](#) required

A reference by OID to the router that this interfa part of.

**secondary-subnets** [net](#)

The list of secondary subnets (and interface addre for this interface.

**state** [bool](#)

The enabled/disabled state of this interface.

## router-type

router-type - Displays the available router types

### Synopsis

```
router-type list
router-type listnames
```

### Description

The router-type command displays the available router types.

### Examples

### Status

### See Also

[router](#)

### Attributes

**description** [string](#)

The description of this router type.

**java-class-name** [string](#) required

The java class name for an implementation of com.cisco.cnr.ricsrv.IRouter that can communicate with this type of router.

**manufacturer** [string](#)

The name of the manufacturer of this type of router.

**name** [string](#) required,unique

The name of this router type.

**router-os-version** [string](#)

The OS version, as a string for this type of router.

## save

save - Saves the current changes to the cluster

### Synopsis

```
save
```

### Description

The save command saves the current configuration changes to the database.

### Examples

### Status

### See Also

## scope

scope - Specifies the scope's properties

### Synopsis

```
scope list
scope listnames
scope <name> create <address> <mask> [template=<template-name>]
    [<attribute>=<value>...]
scope <name> delete
scope <name> set <attribute>=<value> [<attribute>=<value> ...]
scope <name> get <attribute>
scope <name> unset <attribute>
scope <name> disable <attribute>
scope <name> enable <attribute>
scope <name> show

scope <name> listLeases

scope <name> addRange <start> <end>
scope <name> removeRange <start> <end>
scope <name> listRanges

scope <name> addReservation <ipaddr> (<macaddr>|<lookup-key>)
    [-mac|-blob|-string]
```

```

scope <name> removeReservation (<ipaddr>|<macaddr>|<lookup-key>)
                                     [-mac|-blob|-string]
scope <name> listReservations

scope <name> changeMask <mask>
scope <name> clearUnavailable

scope <name> applyTemplate <template-name>

```

## Description

The `scope` command lets you manipulate address ranges in the DHCP server.

When creating a scope using a template, specify `-` for the `<name>` to allow the scope template's `scope-name` to name the scope.

### **scope** <name> **listLeases**

The `listLeases` command lists the leases associated with this scope.

```

scope <name> addRange <start> <end>
scope <name> removeRange <start> <end>
scope <name> listRanges

```

The `addRange` command adds a range of available addresses to the scope. The `<start>` and `<end>` values can either be full IP addresses or host numbers within the scope's subnet. It is an error if either `<start>` or `<end>` is an IP address outside the scope's subnet or is a host number that exceeds the bits allocated to host numbers by the scope's netmask.

The `removeRange` command removes a range of addresses from the scope's control.

The `listRanges` command lists the current ranges of addresses available for allocation.

```

scope <name> addReservation <ipaddr> (<macaddr>|<lookup-key>)
                                     [-mac|-blob|-string]
scope <name> removeReservation (<ipaddr>|<macaddr>|<lookup-key>)
                                     [-mac|-blob|-string]
scope <name> listReservations

```

The `addReservation` command adds a reservation for a specific IP address to a specific MAC address or lookup key.

The `removeReservation` command removes the reservation for an IP address, MAC address, or lookup key.

The `listReservations` command lists the available reservations.

### **scope** <name> **changeMask** <mask>

The `changeMask` command changes the scope's netmask. It also changes the `primary-mask` properties of any secondary scopes. Warnings are issued if the netmask is "narrowed" and any reservations or ranges fall outside the network

defined by the new mask and existing address.

**scope <name> clearUnavailable**

The clearUnavailable command moves all unavailable leases in the specified scope to available.

**scope applyTemplate**

The applyTemplate command applies the specified scope-template to the scope. All properties configured on the scope-template are applied to the scope.

**Note:** The scope command manages a VPN through a virtual attribute:

`vpn [] (AT_STRING, Optional, default: <none>)`

Use this attribute to set or get the VPN ID by VPN name instead of by ID.

## Examples

```
nrcmd> scope scope-example create 192.168.0.0 255.255.255.224
nrcmd> scope scope-example addRange 192.168.0.33 192.168.0.62
nrcmd> scope scope-example addReservation 192.168.0.40 00:d0:ba:d3:bd:3b
nrcmd> scope scope-example changeMask 255.255.255.192
```

## Status

## See Also

[scope-template](#)

## Attributes

### **allocate-first-available** `bool` default = false

Enables you to force the allocation of new IP addresses from this scope to be the first available IP address; otherwise, the default of the 'least recently used' IP address is used.

If this attribute is not set, or is unset, then the DHCP scope attribute priority-address-allocation controls whether to allocate the first available IP address.

If priority-address-allocation is set, and allocate-first-available for the scope is unset, then the scope allocates addresses as if allocate-first-available was set.

If allocate-first-available is enabled or disabled for a scope, then for that scope the setting of priority-address-allocation has no meaning.

### **allocation-priority** `int` default = 0

Assigns an order to scopes for allocating IP addresses. Across scopes, with the highest allocation priority, grant IP addresses until the addresses are exhausted.

You can mix scopes with an allocation-priority along with scopes without a priority in the same network. In this case, scopes with allocation priorities are examined for acceptability before scopes with no allocation-priority. Lower numeric values indicate higher priority.

higher priorities, but an allocation-priority of 0 (the default) means no priority.

If this attribute is not set, or is unset or 0, then the DHCP priority-address-allocation attribute controls the priority of the scope. If the DHCP priority-address-allocation attribute is set, and allocation-priority for the scope is unset, then allocation-priority for the scope is the network number of the scope.

If you explicitly set allocation-priority, then, for that scope, the DHCP setting of priority-address-allocation has no meaning.

### **backup-pct** `percent`

Determines the percentage of available addresses that the server sends to the backup server. If you define this value using the `scope` command, make sure you define it on the main server. If you define it on a backup server, it is ignored.

Used with the `scope` command, the `backup-pct` attribute overrides the defined values on the failover pair for `backup-pct` and `dynamic-bootp-backup-pct`. The attribute value defined with the `scope` command becomes the value used for this scope, whether or not the scope supports `dynamic-bootp`.

If you set the value to zero (0), the backup server receives no addresses. Since 0 is a significant value, once you set it to zero you must unset it for the scope to use the failover pair's `backup-pct` or `dynamic-bootp-backup-pct`.

Note: If the failover pair is configured to use load balancing, `backup-pct` is ignored and 50% is used.

### **bootp** `bool` default = disabled

Controls whether the server accepts BOOTP requests. If you want to allow clients to always receive the same addresses, you must reserve those IP addresses for all your BOOTP clients.

### **deactivated** `bool`

Controls whether a scope extends leases to any clients. A scope that is deactivated does not extend leases to any clients. It treats all IP addresses in its ranges as if they were individually deactivated.

### **description** `string`

Describes the scope.

### **dhcp** `bool` default = enabled

Controls whether the DHCP server accepts DHCP requests for the scope. Disable DHCP if you want a scope to use BOOTP exclusively or you want to deactivate the scope temporarily.

### **dns-host-bytes** `rangeint(1-4)`

Tells DHCP how many bytes in a lease IP address to use when forming reverse in-addr.arpa names. The server forms reverse names by prepending `dns-host-bytes` of IP address (in reverse order) to the reverse zone name. If unset, the server synthesizes an appropriate value based on the scope's subnet size.

### **dynamic-bootp** `bool` default = disabled

Controls whether the server will accept dynamic BOOTP requests for this scope. Dynamic BOOTP requests are BOOTP requests that do not match a reservation, but could be satisfied from the available lease pool. To use this feature you must also enable `bootp`.

### **embedded-policy** `obj(0)`

Displays the embedded policy for a scope.

### **failover-backup-allocation-boundary** `ipaddr`

Sets the IP address allocation boundary for a backup server failover relationship.

If the `allocate-first-available` attribute is set, the backup server allocates IP addresses in descending order from this boundary.

If the `allocate-first-available` attribute is unset or set to false, the boundary used for allocating addresses is half the distance between the first and last IP address configured in the scope. If no IP addresses are available below this boundary, the first IP address available above this boundary is used.

**free-address-config** [nameref](#)(AddrTrapConfig)

Identifies which trap captures unexpected free address events on this scope.

**ignore-declines** [bool](#) default = false

Determines whether the server reacts to server DHCPDECLINE messages that refer to one of the scope's IP addresses.

If enabled (true), the DHCP server ignores all declines that refer to an IP address in this scope. If disabled, the DHCP server sets to UNAVAILABLE every IP address referred to in this scope. Default is false (disabled).

**ping-clients** [bool](#) default = disabled

Controls whether the server should attempt to ping an address before offering a lease. If enabled (true), this attribute indicates a ping timeout.

**ping-timeout** [int](#) default = 300

Sets the number of milliseconds the DHCP server waits for responses. If you make this value too large, you slow down the lease offering processes. If you make this value too small, you reduce the effectiveness of pinging addresses before offering them. Three hundred milliseconds (the default value) is often the best choice.

**policy** [nameref](#)(Policy) default = default, required

Identifies the name of the policy associated with this scope. If not set, the default policy is used. This means that the scope uses all properties set in the default policy (including the lease duration) unless you specifically reset a property.

**primary-subnet** [subnet](#)

Determines the subnet address and mask of the primary scope. Use this attribute when multiple logical IP subnets are present on the same physical network.

**renew-only** [bool](#)

Controls whether to allow existing clients to reacquire their leases, but not offer any leases to new clients. Note that the `renew-only` scope does not change the client associated with its leases (other than to allow a client, currently using the server believes is an available IP address, to continue to use the address).

**selection-tag-list** [string](#)

Associates a comma-separated list of selection tags with a scope. The scope compares a client's selection criteria to the list in order to determine whether the client can obtain an address from the scope.

**subnet** [subnet](#) required

The network address of the IP subnet that this scope represents.

**vpn-id** [int](#) default = 0, immutable

Displays the identifier of the DHCP VPN that contains the in this scope. Define this value with the `vpn vpn-name c` command. Once set, you cannot change this value.

## scope-policy

`scope-policy` - Adds DHCP policy information to a scope

### Synopsis

```
scope-policy <scope-name> delete
scope-policy <scope-name> set <attribute>=<value> [<attribute>=<value> ...]
scope-policy <scope-name> get <attribute>
scope-policy <scope-name> unset <attribute>

scope-policy <scope-name> disable <attribute>
scope-policy <scope-name> enable <attribute>
scope-policy <scope-name> show

scope-policy <scope-name> setLeaseTime <time-val>
scope-policy <scope-name> getLeaseTime

scope-policy <scope-name> setOption <opt-name | id> <value>
scope-policy <scope-name> getOption <opt-name | id>
scope-policy <scope-name> unsetOption <opt-name | id>
scope-policy <scope-name> listOptions

scope-policy <scope-name>
    setVendorOption <opt-name | id> <opt-set-name> <value>
scope-policy <scope-name>
    getVendorOption <opt-name | id> <opt-set-name>
scope-policy <scope-name>
    unsetVendorOption <opt-name | id> <opt-set-name>
scope-policy <scope-name> listVendorOptions
```

### Description

The `scope-policy` command lets you configure embedded DHCP policy information for a DHCP scope. An embedded policy is a collection of DHCP option values and settings associated with (and named by) another object -- in this case a scope. A scope policy is created implicitly when you first reference it, and is deleted when the scope is deleted.

You can set individual option values with the `setOption` command, unset option values with the `unsetOption` command, and view option values with the `getOption` and `listOptions` commands. When you set an option value the DHCP server replaces any existing value or creates a new one as needed for the given option name.

You can use the `setLeaseTime` command to set the values of lease times and the `getLeaseTime` command to display the value of a lease time.

### Examples

```
nrcmd> scope-policy scope-example set backup-pct=30
nrcmd> scope-policy scope-example enable allocate-first-available
```

## Status

## See Also

[policy](#), [client-policy](#), [client-class-policy](#)

## Attributes

### **affinity-period time**

Associates a lease in the AVAILABLE state with the client last held the lease. If the client requests a lease during the affinity period, it is granted the same lease; that is, renewals are prohibited, then it is explicitly not given. Because of the vast IPv6 address space and depending on the generation technique, it could be millions of years before an address ever needs reassignment to a different client, so there is no reason to hold on to this information for that long. To prohibit renewals, enable either the `inhibit-all-renewals` or the `inhibit-renews-at-reboot` attribute.

### **allow-client-a-record-update bool default = disabled**

Determines if a client is allowed to update A records. If the client sets the `flags` in the `FQDN` option to indicate it wants to do the A record update in the request, and the `allow-client-a-record-update` value is `TRUE`, the server allows the client to do the A record update; otherwise, based on other server configurations, the server does the A record update.

### **allow-dual-zone-dns-update bool default = disabled**

Enables DHCP clients to perform DNS updates into two DNS zones. To support these clients, you can configure the DHCP server to allow the client to perform an update, but also to perform an update on the client's behalf.

### **allow-lease-time-override bool default = disabled**

Gives the server control over the lease period. Although a client can request a specific lease time, the server need not honor the request if this attribute is set to `false` (the default). Even if set to `true`, clients can request only a lease time shorter than those configured for the server.

### **allow-non-temporary-addresses bool default = true**

Determines whether DHCPv6 clients can request non-temporary (IA\_NA) addresses.

The default is to allow clients to request non-temporary addresses.

### **allow-rapid-commit bool default = false**

Determines whether DHCPv6 clients can use a Solicit with Rapid Commit option to obtain configuration information with fewer messages. To permit this, make sure that a single server is servicing clients.

This attribute has special handling during the policy processing when checking the Prefix policies (embedded in the Prefixes on a Link). The Prefixes for the Link are processed in alphabetic (case blind) order. For each Prefix embedded and then named policy are checked. Only Prefixes the client has access (based on selection tags, etc.) are checked.

- If any of the prefix policies has this attribute set to `FALSE`, Rapid Commit is not allowed.

- If at least one has it set to TRUE, Rapid Commit is
- Otherwise, the remaining policies in the hierarchy checked.

The default is not to allow clients to use Rapid Commit

**allow-temporary-addresses** [bool](#) default = true

Determines whether DHCPv6 clients can request temporary addresses.

The default is to allow clients to request temporary ad

**default-prefix-length** [rangeint\(0-128\)](#) default = 64

For delegation, specifies the default length of the del if a router (client) does not explicitly request it.

The default length must always be less than or equal to length of the prefix range.

**forward-dnsupdate** [nameref\(DnsUpdateConfig\)](#)

Specifies the name of the update configuration that det which forward zones to include in updates.

**forward-zone-name** [dname](#)

Designates an optional forward zone for DNS updates.

**giaddr-as-server-id** [bool](#) default = false

Enables the DHCP server to set the server-id option on and a DHCPACK to the giaddr of the incoming packet, ins IP address of the server (the default action).

This causes all unicast renews to be sent to the relay of directly to the DHCP server, and so renews arrive at server with option-82 information appended to the packe Some relay agents may not support this capability and, complex configurations, the giaddr might not actually b to which the DHCP client can send A unicast packet. In the DHCP client cannot renew a lease, and must always p rebind operation (where the DHCP client broadcasts a re of sending a unicast to what it believes is the DHCP se

**grace-period** [time](#) default = 5m

Defines the length of time between the expiration of a and the time it is made available for reassignment.

**inhibit-all-renews** [bool](#) default = false

Causes the server to reject all renewal requests, forci to obtain a different address any time it contacts the

**inhibit-renews-at-reboot** [bool](#) default = false

Permits clients to renew their leases, but the server f them to obtain new addresses each time they reboot.

**limitation-count** [int](#)

Specifies the maximum number of clients with the same l that are allowed to have currently active and valid lea

**offer-timeout** [time](#) default = 2m

Instructs the server to wait a specified amount of time has offered a lease to a client, but the offer is not y At the end of the specified time interval, the server m lease available again.

**packet-file-name** [string](#)

Identifies the boot-file to use in the boot process of The server returns this file name in the 'file' field o The packet-file-name cannot be longer than 128 characte

**packet-server-name** [string](#)

Identifies the host-name of the server to use in a client process. The server returns this file name in the 'sname' of its replies. The packet-server-name field cannot be more than 64 characters.

**packet-siaddr** **ipaddr**

Identifies the IP address of the next server in the client process. For example, this might be the address of a TFTP used by BOOTP clients. The server returns this address in the 'siaddr' field of its replies.

**permanent-leases** **bool** default = disabled

Indicates whether leases using this policy are permanent to requesting clients. If leases are permanently granted, it ignores the configured dhcp-lease-time option value and maximum lease time.

**preferred-lifetime** **time** default = 1w

Assigns the default and maximum preferred lifetime for DHCPv6 client interfaces. Expressed in seconds and relative to the time the server sent the packet, this attribute sets the length of time that the address is preferred; that is, unrestricted. When the preferred lifetime expires, the address becomes deprecated and its use is restricted.

**reconfigure** **enumint**(allow=1, disallow=2, require=3) default = allow

Controls DHCPv6 client reconfiguration support:

- |   |          |                                                                                                                                                               |
|---|----------|---------------------------------------------------------------------------------------------------------------------------------------------------------------|
| 1 | allow    | Allows clients to request reconfiguration support and the server will honor the request (default).                                                            |
| 2 | disallow | Allows clients to request reconfiguration support but the server will not honor the clients' request.                                                         |
| 3 | require  | Requires clients to request reconfiguration support and the server drops client Solicit and Request messages that do not include a Reconfigure-Accept option. |

This attribute has special handling during the policy hierarchy processing when checking the Prefix policies (embedded for the Prefixes on a Link. The Prefixes for the Link are processed in alphabetic (case blind) order. For each Prefix embedded and then named policy are checked. Only Prefixes the client has access (based on selection tags, etc.) are checked as follows:

- If any of the prefix policies has this attribute set to disallow or require, that setting is used.
- Otherwise, if at least one has it set to allow, Reconfigure is allowed.
- If no prefix policies have this attribute set, the policies in the hierarchy are checked.

**reconfigure-via-relay** **bool** default = false

Controls whether the server should prefer unicasting or relaying DHCPv6 Reconfigure messages.

If false (the default), the server prefers to unicast Reconfigure messages if the client has one or more valid statefully assigned addresses.

If true, the server prefers to send Reconfigure message via the relay agent unless no relay agent information is available.

Note: When you use this attribute, consider that:

- In networks where the DHCPv6 server cannot communicate

directly with its client devices?for example, wh firewalls are in place?set this value to true.

- The DHCPv6 server does not use embedded and name policies configured on a client when it evaluate this attribute.
- The relay agent cannot be used if the Relay-Forw came from a link-local address.

### **reverse-dnsupdate** [nameref](#)(DnsUpdateConfig)

Specifies the name of the update configuration that det which reverse zones to include in a DNS update.

### **server-lease-time** [time](#)

Tells the server how long a lease is valid. For more f communication with a client, you might have the server leases as leased for a longer period than the client co This also provides more lease-time stability. This valu unless it is longer than the lease time in the dhcp-lea found through the normal traversal of policies.

### **split-lease-times** [bool](#) default = disabled

Specifies a value that the DHCP server might use intern affect lease times.

If enabled, the DHCP server still offers clients lease reflect the configured lease-time option from the appro policy; but the server bases its decisions regarding ex on the 'server-lease-time' value.

### **unavailable-timeout** [time](#) default = 24h

Permits the server to make a lease unavailable for the and then to return the lease to available state. If the configured in the system\_default\_policy, then the defau 86400 seconds (or 24 hours).

### **use-client-id-for-reservations** [bool](#) default = off

Controls how the server database checks for reserved IP addresses.

By default, the server uses the MAC address of the DHCP key for its database lookup. If this attribute is set (enabled), then the server does the check for reserved addresses using the DHCP client-id, which the client us In cases where the DHCP client does not supply the clie server synthesizes it, and uses that value.

### **v4-bootp-reply-options** [optionid4](#)

Lists the options the server returns to all BOOTP clien

### **v4-reply-options** [optionid4](#)

Lists the options the server returns to all DHCPv4 clie or not the client specifically asks for the option data

### **v6-reply-options** [optionid6](#)

Lists the options that should be returned in any replies to DHCPv6 clients.

This attribute has special handling during the policy h processing when checking the Prefix policies (embedded for the Prefixes on a Link. The Prefixes for the Link a processed in alphabetic (case blind) order. For each Pr embedded and then named policy are checked. Only Prefix the client has access (based on selection tags, etc.) a

### **valid-lifetime** [time](#) default = 2w

Assigns the default and maximum valid lifetime for leas DHCPv6 client interfaces. Expressed in seconds and rela

to the time the server sent the packet, this attribute the length of time that an address remains valid. When period of time expires, the address becomes invalid and unusable. The valid lifetime must be greater than or eq to the preferred lifetime.

## scope-template

scope-template - Configures a scope template

### Synopsis

```
scope-template <name> create [<attribute>=<value> ...]
scope-template <name> delete
scope-template <name> set <attribute>=<value> [<attribute>=<value> ...]
scope-template <name> get <attribute>
scope-template <name> unset <attribute>

scope-template <name> disable <attribute>
scope-template <name> enable <attribute>
scope-template <name> show

scope-template <name> create clone=<clone-name>
scope-template <name> apply-to <all | <scope1>[,...]>
```

### Description

The scope-template command lets you configure a template to use when creating scopes.

### Examples

```
nrcmd> scope-template scope-example createscope-template scope-example set ba
```

### Status

### See Also

[scope](#)

### Attributes

#### allocate-first-available **bool**

This boolean attribute forc this scope to be made from the default of the "least r If this attribute is not se allocate the first availabl controlled by the DHCP serv If priority-address-allocat for the scope is not set (u addresses as if allocate-fi allocate-first-available ha enabled or disabled) for a priority-address-allocation

#### allocation-priority **int**

You can use the allocation- such that allocation of IP scopes with a higher priori

scopes are exhausted. An a having an allocation-proior allocation-priority along w (or with an allocation-prio same network. In this case will be examined for accept allocation-priority (or an If this attribute is not se the scope will be controlle priority-address-allocation (and allocation-priority fo allocation-priority for the scope. If allocation-prior scope, then for that scope has no meaning.

**backup-pct** percent

The percentage of available should send to the backup s must be defined for the sco defined in a backup server, of configurations). If it override the failover pair' and dynamic-bootp-backup-pc the value that will be used this scope supports dynamic no addresses will be sent t is a significant value, onc unset in order for this sco values for backup-pct or dy Note: If the failover pair the percentage is ignored a

**bootp** bool

Controls whether the server for this scope. If you want same addresses, you need to BOOTP clients.

**deactivated** bool

A deactivated scope will no all of the addresses in its deactivated.

**description** string

Describes the scope templat

**dhcp** bool

Controls whether the server for this scope. Typically, bootp is enabled for that s to be used exclusively for

**dns-host-bytes** rangeint(1-4)

This value tells DHCP how m use when forming in-addr.ar in-addr zone by prepending order) to the reverse zone If this is unset, the serve based on the scope's subnet

**dynamic-bootp** bool

Controls whether the server

requests for this scope. D  
that do not match a reserva  
available lease pool. To us

**embedded-policy** [obj\(0\)](#)

The embedded policy object

**free-address-config** [nameref\(AddrTrapConfig\)](#)

The free-address trap confi

**grace-period** [time](#)

The length of time between  
is made available for re-as

**ignore-declines** [bool](#)

This attribute controls whe  
DHCPDECLINE request referen  
this attribute is enabled,  
declines which reference an  
attribute is not set, the D  
every IP address which is r  
The default value is false,  
processed normally, and the  
DHCPDECLINE messages are se

**name** [string](#) required,unique

The name of this scope temp

**offer-timeout** [time](#)

If the server offers a leas  
accepted, the server will w  
making the lease 'available

**options-expr** [expr](#)

An expression to define the  
to be created for a scope o

**ping-clients** [bool](#)

Controls whether the server  
addresses before offering l

**ping-timeout** [int](#)

The number of milliseconds  
responses. If you make thi  
lease offering processes. I  
reduce the effectiveness of

**policy** [nameref\(Policy\)](#) default = default

The name of the policy asso

**ranges-expr** [expr](#)

An expression to define the  
for a scope object.

**renew-only** [bool](#)

Controls whether to allow e  
but not offer any leases to  
scope will not change the c  
(other than to allow a clie  
is an available IP address

**router-host** [int](#)

This is use to create the i  
router interface because ro  
It takes ipaddress and mask

**scope-description-expr** `expr`

Defines an AT\_STRING expres on the scope object created

**scope-name** `expr`

An expression to define the when using the scope templa

**selection-tag-list** `string`

The list of selection tags

**update-dns-for-bootp** `bool`

If the server is replying t from a Scope which is confi this property before beginn an administrator to prevent allowing updates for DHCP c 'Update-dns-for-bootp' can command; that global comman as necessary.

**vpn-id** `int` default = 0

The id of the dhcp vpn that vpn-id of a scope must be i is created, and cannot be e

## scope-template-policy

`scope-template-policy` - Edits a DHCP policy embedded in a scope-template

### Synopsis

```

scope-template-policy <name> delete
scope-template-policy <name> set
    <attribute>=<value>
    [<attribute>=<value> ...]
scope-template-policy <name> get <attribute>
scope-template-policy <name> disable <attribute>
scope-template-policy <name> enable <attribute>
scope-template-policy <name> show

scope-template-policy <name> setLeaseTime <time-val>
scope-template-policy <name> getLeaseTime

scope-template-policy <name> setOption <opt-name | id> <value>
scope-template-policy <name> getOption <opt-name | id>
scope-template-policy <name> unsetOption <opt-name | id>
scope-template-policy <name> listOptions

scope-template-policy <name>
    setVendorOption <opt-name | id> <opt-set-name> <value>
scope-template-policy <name>
    getVendorOption <opt-name | id> <opt-set-name>
scope-template-policy <name>
    unsetVendorOption <opt-name | id> <opt-set-name>
scope-template-policy <name> listVendorOptions

```

### Description

The `scope-template-policy` command lets you configure a DHCP policy embedded in a DHCP scope template. An embedded policy is a collection of DHCP option values and settings associated with (and

named by) another object -- in this case a scope template. A scope-template-policy is created implicitly when you first reference it, and is deleted when the scope-template is deleted.

You can set individual option values with the `setOption` command, unset option values with the `unsetOption` command, and view option values with the `getOption` and `listOptions` commands. When you set an option value the DHCP server will replace any existing value or create a new one as needed for the given option name.

## Examples

```
nrcmd> scope-template-policy exampleScope set default-prefix-length=32
nrcmd> scope-template-policy exampleScope enable allow-rapid-commit
```

## Status

## See Also

[policy](#), [client-policy](#), [client-class-policy](#), [dhcp-address-block-policy](#), [link-policy](#), [scope-policy](#)

## Attributes

### **affinity-period** [time](#)

Associates a lease in the AVAILABLE state with the client last held the lease. If the client requests a lease during affinity period, it is granted the same lease; that is, renewals are prohibited, then it is explicitly not given. Because of the vast IPv6 address space and depending on generation technique, it could be millions of years before address ever needs reassignment to a different client, so there is no reason to hold on to this information for that long. To prohibit renewals enable either the `inhibit-all-renewals` or the `inhibit-renews-at-reboot` attribute.

### **allow-client-a-record-update** [bool](#) default = disabled

Determines if a client is allowed to update A records. If the client sets the flags in the FQDN option to indicate it wants to do the A record update in the request, and if the value is TRUE, the server allows the client to do the A record update; otherwise, based on other server configurations, the server does the A record update.

### **allow-dual-zone-dns-update** [bool](#) default = disabled

Enables DHCP clients to perform DNS updates into two DNS zones. To support these clients, you can configure the DHCP server to allow the client to perform an update, but also to perform the update on the client's behalf.

### **allow-lease-time-override** [bool](#) default = disabled

Gives the server control over the lease period. Although a client can request a specific lease time, the server need not honor the request if this attribute is set to false (the default). Even if set to true, clients can request only lease time shorter than those configured for the server.

### **allow-non-temporary-addresses** [bool](#) default = true

Determines whether DHCPv6 clients can request non-temporary (IA\_NA) addresses.

The default is to allow clients to request non-temporary

**allow-rapid-commit** **bool** default = false

Determines whether DHCPv6 clients can use a Solicit with Rapid Commit option to obtain configuration information fewer messages. To permit this, make sure that a single server is servicing clients.

This attribute has special handling during the policy hierarchy processing when checking the Prefix policies (embedded in the Prefixes on a Link. The Prefixes for the Link are processed in alphabetic (case blind) order. For each Prefix embedded and then named policy are checked. Only Prefixes the client has access (based on selection tags, etc.) are

- If any of the prefix policies has this attribute set FALSE, Rapid Commit is not allowed.
- If at least one has it set to TRUE, Rapid Commit is allowed.
- Otherwise, the remaining policies in the hierarchy are checked.

The default is not to allow clients to use Rapid Commit.

**allow-temporary-addresses** **bool** default = true

Determines whether DHCPv6 clients can request temporary addresses.

The default is to allow clients to request temporary addresses.

**default-prefix-length** **rangeint(0-128)** default = 64

For delegation, specifies the default length of the delegation if a router (client) does not explicitly request it.

The default length must always be less than or equal to length of the prefix range.

**forward-dnsupdate** **nameref(DnsUpdateConfig)**

Specifies the name of the update configuration that determines which forward zones to include in updates.

**forward-zone-name** **dname**

Designates an optional forward zone for DNS updates.

**giaddr-as-server-id** **bool** default = false

Enables the DHCP server to set the server-id option on a packet and a DHCPACK to the giaddr of the incoming packet, instead of IP address of the server (the default action).

This causes all unicast renews to be sent to the relay agent instead of directly to the DHCP server, and so renews arrive at the server with option-82 information appended to the packet. Some relay agents may not support this capability and, in complex configurations, the giaddr might not actually be the to which the DHCP client can send a unicast packet. In that case the DHCP client cannot renew a lease, and must always perform a rebinding operation (where the DHCP client broadcasts a request of sending a unicast to what it believes is the DHCP server).

**grace-period** **time** default = 5m

Defines the length of time between the expiration of a lease and the time it is made available for reassignment.

**inhibit-all-renews** **bool** default = false

Causes the server to reject all renewal requests, forcing clients to obtain a different address any time it contacts the server.

**inhibit-renews-at-reboot** **bool** default = false

Permits clients to renew their leases, but the server forbids

them to obtain new addresses each time they reboot.

**limitation-count** `int`

Specifies the maximum number of clients with the same link ID that are allowed to have currently active and valid leases.

**offer-timeout** `time` default = 2m

Instructs the server to wait a specified amount of time before it has offered a lease to a client, but the offer is not yet accepted. At the end of the specified time interval, the server makes the lease available again.

**packet-file-name** `string`

Identifies the boot-file to use in the boot process of a client. The server returns this file name in the 'file' field of its replies. The packet-file-name cannot be longer than 128 characters.

**packet-server-name** `string`

Identifies the host-name of the server to use in a client boot process. The server returns this file name in the 'sname' field of its replies. The packet-server-name field cannot be longer than 64 characters.

**packet-siaddr** `ipaddr`

Identifies the IP address of the next server in the client boot process. For example, this might be the address of a TFTP server used by BOOTP clients. The server returns this address in the 'siaddr' field of its replies.

**permanent-leases** `bool` default = disabled

Indicates whether leases using this policy are permanent to requesting clients. If leases are permanently granted, the server ignores the configured dhcp-lease-time option value and maximum lease time.

**preferred-lifetime** `time` default = 1w

Assigns the default and maximum preferred lifetime for DHCPv6 client interfaces. Expressed in seconds and relative to the time the server sent the packet, this attribute sets the length of time that the address is preferred; that is, the address is unrestricted. When the preferred lifetime expires, the address becomes deprecated and its use is restricted.

**reconfigure** `enumint`(allow=1, disallow=2, require=3) default = allow

Controls DHCPv6 client reconfiguration support:

- 1 allow Allows clients to request reconfiguration support and the server will honor the request (default).
- 2 disallow Allows clients to request reconfiguration support but the server will not honor the clients' request.
- 3 require Requires clients to request reconfiguration support and the server drops client Solicit and Request messages that do not include a Reconfigure-Accept option.

This attribute has special handling during the policy handling process when checking the Prefix policies (embedded in the Prefixes on a Link. The Prefixes for the Link are processed in alphabetic (case blind) order. For each Prefix, the embedded and then named policy is checked. Only Prefixes for which the client has access (based on selection tags, etc.) are checked as follows:

- If any of the prefix policies has this attribute set to

disallow or require, that setting is used.

- Otherwise, if at least one has it set to allow, Rec is allowed.
- If no prefix policies have this attribute set, the policies in the hierarchy are checked.

### **reconfigure-via-relay** **bool** default = false

Controls whether the server should prefer unicasting or relaying DHCPv6 Reconfigure messages.

If false (the default), the server prefers to unicast Reconfigure messages if the client has one or more valid statefully assigned addresses.

If true, the server prefers to send Reconfigure messages via the relay agent unless no relay agent information is available.

Note: When you use this attribute, consider that:

- In networks where the DHCPv6 server cannot communicate directly with its client devices?for example, where firewalls are in place?set this value to true.
- The DHCPv6 server does not use embedded and named policies configured on a client when it evaluates this attribute.
- The relay agent cannot be used if the Relay-Forw came from a link-local address.

### **reverse-dnsupdate** **nameref**(DnsUpdateConfig)

Specifies the name of the update configuration that determines which reverse zones to include in a DNS update.

### **server-lease-time** **time**

Tells the server how long a lease is valid. For more frequent communication with a client, you might have the server consider leases as leased for a longer period than the client considers. This also provides more lease-time stability. This value is used unless it is longer than the lease time in the dhcp-leases found through the normal traversal of policies.

### **split-lease-times** **bool** default = disabled

Specifies a value that the DHCP server might use internally to affect lease times.

If enabled, the DHCP server still offers clients lease times that reflect the configured lease-time option from the appropriate policy; but the server bases its decisions regarding expiration on the 'server-lease-time' value.

### **unavailable-timeout** **time** default = 24h

Permits the server to make a lease unavailable for the specified time and then to return the lease to available state. If the value is configured in the system\_default\_policy, then the default is 86400 seconds (or 24 hours).

### **use-client-id-for-reservations** **bool** default = off

Controls how the server database checks for reserved IP addresses.

By default, the server uses the MAC address of the DHCP client as the key for its database lookup. If this attribute is set to true (enabled), then the server does the check for reserved addresses using the DHCP client-id, which the client supplies. In cases where the DHCP client does not supply the client-id, the server synthesizes it, and uses that value.

### **v4-bootp-reply-options** **optionid4**

Lists the options the server returns to all BOOTP clients.

### **v4-reply-options** [optionid4](#)

Lists the options the server returns to all DHCPv4 client or not the client specifically asks for the option data.

### **v6-reply-options** [optionid6](#)

Lists the options that should be returned in any replies to DHCPv6 clients.

This attribute has special handling during the policy processing when checking the Prefix policies (embedded for the Prefixes on a Link. The Prefixes for the Link are processed in alphabetic (case blind) order. For each Prefix embedded and then named policy are checked. Only Prefixes the client has access (based on selection tags, etc.) are

### **valid-lifetime** [time](#) default = 2w

Assigns the default and maximum valid lifetime for lease DHCPv6 client interfaces. Expressed in seconds and relative to the time the server sent the packet, this attribute is the length of time that an address remains valid. When period of time expires, the address becomes invalid and unusable. The valid lifetime must be greater than or equal to the preferred lifetime.

## **server**

`server` - Configures and controls the server objects

### **Synopsis**

The `server` keyword is optional. You can enter all commands starting with just the server type (`<server>`).

```
server <server> disable <attribute>
```

```
server <server> enable <attribute>
```

```
server <server> get <attribute>
```

```
server <server> start
```

```
server <server> stop
```

```
server <server> reload
```

```
server <server> getHealth
```

```
server <server> getStats
```

```
server <server> setDebug < category-list>=<value> [<output>]
```

```
server <server> unsetDebug
```

```
server <server> getDebug
```

```
server <server> serverLogs show
```

```
server <server> serverLogs nlogs=<nlogs> logsize=<logsize>
```

### **Description**

The `server` command is used to send general control messages and queries to the servers in the cluster. `<server>` may be `ccm`, `dhcp`, `dns`, `snmp`, or `tftp`.

Not all servers support all commands.

The `stop` command stops the specified server.

The `start` command starts the specified server.

The reload command reloads the specified server, possibly with new configuration information.

The getHealth command gets the specified server's current health value.

The getStats command gets the specified server's current statistics.

The setDebug command sets the debugging level and debug message output location. The valid values for <output> are: MLOG, FILE, and WINDOW. The default value is MLOG.

The unsetDebug command turns off debugging output.

The serverLogs show command displays the number of log files and the maximum size for each file.

The serverLogs command allows setting the two server logging parameters, nlogs and logsize. Either or both may be specified in the command, and changes occur only to the one(s) specified. When setting logsize, the value may be suffixed with K or M to signify units of thousands or millions. Note that in order for these changes to take effect you must save the changes and restart the server Agent.

```
server dhcp serverLogs nlogs=6 logsize=500K
server dns serverLogs logsize=5M
```

The features that can be enabled or disabled are:

start-on-reboot

For any server, controls whether this server is started by the Network Registrar server agent. Disabling this feature for a server is useful for clusters that want to provide a single protocol service. By default both the DNS and DHCP services are enabled.

The properties that can be read with the get command are:

version

The version string for this server.

## Examples

```
nrcmd> server DNS stop
nrcmd> server DNS start
nrcmd> server DHCP getHealth
nrcmd> server DNS disable start-on-reboot
```

## See Also

[ccm](#)(nrcmd), [dns](#)(nrcmd), [dhcp](#)(nrcmd), [snmp](#)(nrcmd), [tftp](#)(nrcmd), [nrcmd](#) documentation

## session

session - Configures nrcmd program session parameters

## Synopsis

```
session set <attribute>=<value>
session get <attribute>
session enable <attribute>
session disable <attribute>
```

**session show**

**session cache <refresh | clear>**

**session listNetInterfaces**

**session log [<filename>]**

## Description

The session command lets you view and set session parameters, such as the session visibility and the default output format of your nrcmd program session.

The listNetInterfaces command returns a list of the network interfaces that are present on the machine running Network Registrar. Both IPv4 and IPv6 interfaces are included.

The log command closes the currently open log file, if any, and opens a new log file to which subsequent output is written if a filename is specified.

## Features

### cache

The CLI caches many configuration objects that it reads. If multiple users are making changes simultaneously, one CLI instance may have cached an out-of-date version of an object. The cache refresh command causes the CLI to clear its local cache of all unmodified objects, forcing the CLI to re-read objects from the configuration database. The cache clear command forces the CLI to clear all cached data, whether or not unsaved changes have been made.

### cluster

The name of the current cluster. This is a read-only property.

### current-vpn (or current-namespace)

The name of the current VPN presently in effect. This is a read/write property. This is the VPN used in commands which need a VPN (e.g., lease, report, export) and no VPN is explicitly specified when the command is issued. The reserved vpn-name "global" is the VPN which contains all of the IP addresses not in any explicitly named VPN, and is the default current-vpn for the session.

Note: If the session is interactive, the nrcmd> prompt is changed to nrcmd [VPN=<vpn-name>]> except when the current-vpn is global.

### default-format

The default-format is an enumerated string that can have the following settings. If not set, it is interpreted as script.

user - Show objects in user-friendly form: one property per line.

script - Show objects in script friendly form: one object per line.

#### error

The error detail for the last request to the CCM server that failed. This report is often used to get more detailed information about why a nrcmd command failed to complete successfully.

Note: The error detail may not reflect the final result of the nrcmd command because some commands result in multiple requests to CCM, one or more of which may have failed and the error detail reports only on the last failed operation (which may not be the last operation).

#### groups

Displays the list of groups associated with the current user.

#### roles

Displays the list of roles associated with the current user.

#### scope-edit-mode

The edit mode currently in effect when editing DHCP scopes. The valid values are 'staged' and 'synchronous' (or 'sync'). The value 'default' will use the value configured at the CCM server. This feature only appears in Network Registrar 6.2 and later releases.

#### user-name

The name of the current user. This is a read-only property.

#### version

Retrieves the version associated with the Network Registrar cluster.

#### visibility

The session visibility setting. The default level is 5, which is the normal operating mode. Visibility level 3 can be set to view 'expert' level properties. These additional, reserved properties should be modified with caution. Incorrect settings for these properties can result in network outages. The default settings are the best choice in virtually all cases and should not be changed, except to address a specific network issue. Changes should not be attempted without assistance from the Cisco TAC.

#### zone-edit-mode

The edit mode currently in effect when editing CNR zones. The valid values are 'staged' and 'synchronous' (or 'sync'). The value 'default' will use the value configured at the CCM server. This only appears for CNR 6.2 and later releases.

## Examples

## Status

## See Also

# snmp

snmp - Configures and controls the SNMP server

## Synopsis

```
snmp disable <attribute>
snmp enable <attribute>

snmp get <attribute>
snmp set <attribute>=<value> [<attribute>=<value> ...]
snmp unset <attribute>
snmp [show]
```

## Description

The snmp command configures the Network Registrar SNMP server. The SNMP server makes some Network Registrar statistics available to SNMP clients, and generates SNMP traps based on the status of the other Network Registrar servers.

## Examples

## Status

## See Also

[server](#)

## Attributes

**cache-ttl** **time** default = 60s

Controls how long CNR data can be cached by the SNMP server a responds to SNMP GETs.

**community** **string** default = public

SNMP requests must present a matching community-string in order to be processed by the SNMP server.

**log-settings** **flags**(default=1, no-success-messages=2, incoming-packet-detail=3, outgoing-packet-detail=5, scp-detail=6, snmp-detail=7) default = default

Configures log messages. The server logs more or fewer messages depending on this parameter.

**name** **string** default = CNRSNMP

The name of the server (constant for now).

**server-active** **bool** default = true

If 'true', the server will run when it is started. If 'false' the server will not run when it is started.

**trap-source-addr** **ipaddr**

An optional address to use as the sender address in outgoing trap packets.

# snmp-interface

snmp-interface - Configures the SNMP server's network interfaces

## Synopsis

```
snmp-interface <name> create [<attribute>=<value>] [<attribute>=<value> ...]
```

```
snmp-interface <name> delete
snmp-interface list
snmp-interface listnames
snmp-interface <name> show
snmp-interface <name> set <attribute>=<value> [<attribute>=<value> ...]
snmp-interface <name> get <attribute>
snmp-interface <name> unset <attribute>

snmp-interface <name> enable <attribute>
snmp-interface <name> disable <attribute>
```

## Description

The `snmp-interface` command configures network interfaces for use by the NetworkRegistrar SNMP server. If there are no defined interfaces, the server discovers and uses all available interfaces on the system. When this list is present, the server only uses available interfaces, if any, that match this list.

## Examples

```
nrcmd> snmp-interface example1 create
nrcmd> snmp-interface example1 set address=192.168.0.58
```

## Status

## See Also

## Attributes

### address [subnet](#)

The IP address and subnet mask of an interface that should use.

**name** [string](#) required,unique,immutable

## subnet

`subnet` - Describes a contiguous range of IP address space in the `address-space` model

## Synopsis

```
subnet <address/mask> create
subnet <address> delete
subnet list
subnet listnames
subnet <address> show

subnet <address> get <attribute>
subnet <address> set <attribute>=<value> [<attribute>=<value> ...]
subnet <address> unset <attribute>
```

## Description

A subnet is a correctly aligned, power-of-2 sized sequence of IP

addresses. In the address space model, subnets are leaf nodes while address-blocks are root, internal or leaf nodes of a tree describing the address space.

## Examples

```
nrcmd> subnet 192.168.0.0 create
nrcmd> subnet 192.168.0.64 set primary-subnet=192.168.0.0
nrcmd> subnet 192.168.0.64 show primary-subnet
```

## Status

## See Also

[address-block](#)

## Attributes

### address [subnet](#) required,immutable

Sets the subnet address for this subnet. Stored as a network address and a mask determining how large the range is.

### description [string](#)

Provides a description of this subnet.

### dns-host-bytes [rangeint](#)(1-4)

Determines the correct in-addr.arpa name to create, when creating reverse zones from a subnet. Based on this value and the subnet size, either a new reverse zone is created, or delegation records are put into the parent reverse zone. If unset, the server synthesizes an appropriate value based the subnet size. This value parallels the dns-host-bytes attribute in the Scope class.

### failoverpair [oid](#)

For DHCP allocation, assigns a subnet to a CCMFailoverPair or to a single CCMCluster object.

### forward-zone-name [dname](#)

Names the forward zone associated with this subnet.

### interface [oid](#)

Assigns a subnet to a router interface.

### owner [oid](#)

Identifies the owner of this object, referenced by OID. Use owner field to group similarly owned objects and to limit access to objects by their owner.

If this attribute is unset, the object is assumed to have the same owner as its parent.

### parent [oid](#)

Identifies the parent address block for this subnet.

### primary-subnet [subnet](#)

Contains the subnet number of a subnet's primary subnet. Use this attribute when multiple logical IP subnets are present on the same physical network.

### region [oid](#)

Associates an object with a region. The object is referenced by its OID. Use the region field to group similarly located objects to limit access to objects by their region.

If this attribute is unset, the object is assumed to have the same region as its parent.

**reverse-zone-name** [dname](#)

Names the reverse zone associated with this subnet.

**subnet-state** [flags](#)(ric-allocated=1, dhcp-allocated=2, reclaiming=3) default =

Shows the current state of this subnet.

**type** [nameref](#)(CCMAAddrSpaceType)

Identifies the type of this subnet. This attribute contains name of a CCMAAddrSpaceType object that contains more information about the type.

If this attribute is unset, the object is assumed to have the same type as its parent.

**vpn-id** [int](#) default = 0, immutable

Specifies the VPN that contains the subnet address for this subnet.

## sync-from-dns

sync-from-dns - Synchronizes CCM from DNS

### Synopsis

`sync-from-dns`

### Description

The `sync-from-dns` command rebuilds CCM management databases from the DNS server. Live server RR data may have the potential, over time, to go out of synchronization with the managed host data, particularly if live RRs are managed both locally and regionally. The `sync-from-dns` command can be used to perform synchronization of DNS data from the DNS server(s) to the CCM management database(s).

It is only available when NRCMD is at visibility 3.

### Examples

### Status

### See Also

[session](#)

## tftp

tftp - Configures and controls the TFTP server

### Synopsis

`tftp disable` <attribute>

`tftp enable` <attribute>

`tftp get` <attribute>

`tftp set` <attribute>=<value> [=<value> ...]

`tftp unset` <attribute>

`tftp show`

`tftp serverLogs show`

```
tftp serverLogs nlogs=<nlogs> logsize=<logsize>
```

## Description

The tftp command lets you configure the TFTP server in the cluster.

The serverLogs show command displays the number of log files and the maximum size for each file.

The serverLogs command allows you to set the two server logging parameters, nlogs and logsize. You can set one parameter or both. Changes occur only to the one or ones specified. When setting logsize, you can add the suffix K or M signify units of thousands or millions. Note that in order for these changes to take effect you must save the changes and restart the server Agent.

```
tftp serverLogs nlogs=6 logsize=500K
tftp serverLogs logsize=5M
```

## Examples

```
nrcmd> tftp enable docsis-access
nrcmd> tftp set file-cache-directory="CacheDir"
nrcmd> tftp reload
```

## Status

## See Also

[server](#)(nrcmd)

## Attributes

### active-directory-domain [string](#) required

Specifies the name of an Active Directory Domain the TFTP server will use to provide dynamic configuration file support.

### csrc-configuration-file [string](#) required

Specifies the path to a configuration file the TFTP server uses when loading the CSRC library. The CSRC library enables the server to generate DOCSIS modem configuration files. The path of the CSRC configuration file is typically /CSRC\_INSTALL\_DIR/conf/csrm.cfg.

### default-device [string](#) required

Specifies the name of the default disk device the TFTP server uses when none is specified in the pathname contained in the request. This property is specifically to be used on NT to specify a default drive letter.

### file-cache [bool](#) default = disabled

Specifies whether the TFTP server should perform file caching of files located in the file-cache-directory.

### file-cache-directory [string](#)

Specifies a path to a cache directory the TFTP server will find the files to put into cache. Upon start up the TFTP server will load all the files located in this directory into cache.

### file-cache-max-memory-size [rangeint](#)(0-2147483647) default = 32000

Specifies the maximum number of bytes available to the server.

for file-caching.  
Valid range: 0-2147483647

**home-directory** [string](#) required

Specifies a path to a home directory the TFTP server will use to resolve TFTP requests. If use-home-directory-as-root is disabled, the home directory is used in conjunction with the paths specified in the search-list to resolve TFTP requests.

**initial-packet-timeout** [rangeint\(1-2147483647\)](#) default = 5, required

Specifies the initial length of time the TFTP server will wait after sending a response to a client before declaring that the request timed-out and sending a retransmission to the client.

**ldap-host-name** [string](#) default = localhost, required

Specifies the hostname or IP address of an LDAP server the TFTP server will use to provide dynamic configuration file support.

**ldap-initial-timeout** [rangeint\(1-2147483647\)](#) default = 10, required

Specifies the initial length of time the TFTP server will wait after sending a request to an LDAP server before declaring the request timed-out and sending a retransmission to the server.

**ldap-maximum-timeout** [rangeint\(1-2147483647\)](#) default = 60, required

Specifies the maximum length of time the TFTP server will wait after transmitting the initial LDAP request before giving up and retrying on that request.

**ldap-password** [string](#) required

Specifies the password the TFTP server will use when connecting to an LDAP server.

**ldap-port-number** [rangeint\(1-65535\)](#) default = 389, required

Specifies the port number the TFTP server will use when communicating with an LDAP server.

**ldap-root-dn** [string](#) required

Specifies the root distinguished name the TFTP server will use to locate the root of the directory tree for dynamic configuration file support.

**ldap-use-ssl** [bool](#) default = disabled, required

Specifies whether or not the TFTP server will use SSL when communicating with an LDAP server. If this feature is disabled, the TFTP server will not use SSL when communicating with LDAP.

**ldap-user-name** [string](#) required

Specifies the user name the TFTP server will use when connecting to an LDAP server.

**log-level** [rangeint\(0-4\)](#) default = 3, required

Specifies the level of verbosity the TFTP server will employ when writing log messages to the TFTP server log file. Each integer value from 0 through 4 enables the following log levels: None, Error, Warning, Information and Activity.

**log-settings** [flags\(verbose=1, no-success-messages=2\)](#) default = verbose

Enables the TFTP server to allow control over additional details about the events listed in the log-settings. These details help analyze a problem, but can also cause the log files to fill up quickly if left enabled for a long period of time. When log files fill up quickly, they also turn over frequently. So you might lose important information. By default, the TFTP server's logging is verbose.

The possible flags are:

- 1 verbose

Causes all the logging in the server to be active. A success message is printed to the log for each successful transfer.

2 no-success-messages

Causes the single line message that is normally logged for every successful read from or write to the TFTP server to not appear. It affects logging only for successful file reads from the TFTP server.

**max-inbound-file-size** **string** default = 1024k, required

Specifies the maximum file size limit that the TFTP server enforces for a file written to the TFTP server. Default unlimited. Use k, m or g to indicate kilobytes, megabytes or gigabytes.

**min-socket-buffer-size** **rangeint**(1-2147483647) default = 65536, required

Specifies the minimum socket buffer size the TFTP server will use for the well known port on which it is listening for TFTP requests.

**packet-trace-level** **rangeint**(0-4) default = 0, required

Specifies the level of verbosity the TFTP server will employ when writing messages to the server trace file. Each integer value from 1 through 4 enables increasing levels of tracing. Setting packet trace level to 0 disables tracing.

**port-number** **rangeint**(1-65535) default = 69, required

Specifies the UDP port number the TFTP server will use to listen for TFTP requests.

**read-access** **bool** default = enabled, required

Specifies how the TFTP server should respond to file read requests from TFTP clients. If this feature is disabled, the TFTP server will refuse file read requests.

**search-list** **string** required

Specifies a comma separated list of paths the TFTP server will use to resolve TFTP requests. If use-home-directory-as-root is enabled, the paths in the search list are ignored and the specified home directory is used to resolve all TFTP requests.

**session-timeout** **rangeint**(1-2147483647) default = 20, required

Specifies the maximum length of time the TFTP server will wait after transmitting the initial response before giving up on that response. If no response is received from the client within this timeout period, the TFTP session is terminated.

**use-home-directory-as-root** **bool** default = disabled, required

Specifies whether or not the TFTP server will treat pathnames contained within TFTP requests as if the paths were rooted at the specified home directory. If this feature is enabled, the server will attempt to resolve both absolute and relative pathnames to paths located beneath the specified home directory.

**write-access** **bool** default = disabled, required

Specifies how the TFTP server should respond to file write requests from TFTP clients. If this feature is disabled, the TFTP server will refuse file write requests.

**write-allow-file-create** **bool** default = disabled

Specifies whether to allow file creation on a PUT. If false and write-access equals true, the file must exist on the server for a PUT to succeed. If true and write-access is true, then the file does not need to exist on the server first and will be created on a PUT. max-inbound-file-size will apply a limit to the size of the file.

the file.

## tftp-interface

tftp-interface - Configures the network interfaces of the TFTP server

### Synopsis

```
tftp-interface <name> create [<attribute>=<value>]
tftp-interface <name> delete
tftp-interface list
tftp-interface listnames
tftp-interface <name> show
tftp-interface <name> set <attribute>=<value> [<attribute>=<value> ...]
tftp-interface <name> get <attribute>
tftp-interface <name> unset <attribute>

tftp-interface <name> enable <attribute>
tftp-interface <name> disable <attribute>
```

### Description

The tftp-interface command configures network interfaces for use by the Network Registrar TFTP server. The TFTP interface logically represents the hardware interface (for example, an Ethernet or Token Ring network interface card) that the TFTP server uses. The TFTP server uses the configured address information to determine which interface to use to send and receive packets.

If there are no defined interfaces, the server discovers and uses all available interfaces on the system. When this list is present, the server only uses the available interfaces, if any, that match this list.

### Examples

### Status

### See Also

### Attributes

#### address [subnet](#)

The IP address and subnet mask of an interface that the server should use.

#### ip6address [prefix](#)

Specifies the IPv6 address and prefix length of one or more interfaces that the TFTP server should use.

#### name [string](#) required,unique,immutable

A user-assigned name for the interface.

## trap-recipient

trap-recipient - Configures destinations for SNMP trap messages

## Synopsis

```
trap-recipient <name> create
trap-recipient <name> delete
trap-recipient <name> enable <attribute>
trap-recipient <name> disable <attribute>
trap-recipient <name> set <attribute>=<value> [<attribute>=<value> ...]
trap-recipient <name> unset <attribute>
trap-recipient <name> get <attribute>
trap-recipient <name> [show]
trap-recipient list
```

## Description

The `trap-recipient` command configures management stations to which the Network Registrar SNMP server sends trap messages. The traps to be generated are set on the DNS and DHCP server traps-enabled attribute.

## Examples

```
nrcmd> trap-recipient example-recipient createtrap-recipient example-recipient
```

## Status

## See Also

[dhcp](#), [dns](#), [addr-trap](#)

## Attributes

### **agent-addr** [ipaddr](#)

An IP address to use as the source agent-addr of this recipient.

### **community** [string](#)

The SNMP community string of this trap recipient.

### **ip-addr** [ipaddr](#)

Specifies the IP address of this trap recipient.

### **port-number** [int](#) default = 162

The optional IP port number of this trap recipient.

# update-policy

`update-policy` - Configures DNS update policies.

## Synopsis

```
update-policy <name> create
update-policy <name> delete
update-policy list
update-policy listnames
update-policy <name> show
update-policy <name> get <attribute>
update-policy <name> unset <attribute>

update-policy <name> rules add "<value>" [<index>]
update-policy <name> rules remove <index>
```

## Description

The `update-policy` command lets you configure DNS update-policies. The most significant property of an update policy is an ordered list of rules. The rules are used to restrict or permit updates to DNS names. When adding a new rule, enclose the complete string in quotation marks. Use the backslash (`\`) to allow square brackets (`[ ]`) in the rule.

Note: If an update ACL has been configured on the zone, any `update-policy` configuration is ignored.

## Examples

```
nrcmd> update-policy test create
nrcmd> update-policy test rules add "grant any wildcard example* SRV"
nrcmd> update-policy test rules add "deny 1.1.1.1 wildcard \[a-z\]* A"
```

## Status

## See Also

## Attributes

**name** [string](#) required,unique,immutable

Specifies the name of the Update Policy.

**rules** [rule](#)

Lists rules that make up the update policy. Each rule has the following syntax:

**action:** Can be `grant` or `deny`.  
    `grant` - will allow an update if the rest of the rule matches.  
    `deny` - will deny an update if the rest of the rule matches.

**acl-list:** A list of one or more ip addresses, network addresses, keys and/or named acl references. Note key names must be prefixed with "key " (i.e. "key key.example" )

**keyword:** Can be `name`, `subdomain` or `wildcard`.  
    `name` - used to specify a specific RR.  
    `subdomain` - used to specify a subdomain name.  
    `wildcard` - used to specify a name with wildcard characters.

**value:** The name, subdomain or wildcard value associated with the specified keyword. Note that all values are relative to the zone in which they are applied. Therefore it is not necessary to add the zone name to the end of the value.

The supported wildcard characters are:

- \* Will match zero or more characters. For example, the pattern `dhcp-*` matches all strings with the `dhcp-` prefix including the string `dhcp-`.
- ? Will match a single character. For example, the pattern `zone?.com` matches `zone1.com`, `zone2.com`, etc but does not match `zone.com`.

[...] Will match any characters listed within brackets. For example, you can provide a pattern such as 0-9 or a-z. If the pattern also contains the - character, make it the first character in the list (i.e. dhcp[-a-z]\*)

rr-types: A comma delimited list of RR types for this rr-type can also be negated using the exclamation point (i.e. !A,!TXT). You can also specify all types with an asterisk (\*).

## vpn

vpn - Defines a logical VPN within which other DHCP objects may be configured

Note: The namespace command is a synonym for compatibility with earlier versions of Network Registrar.

### Synopsis

```
vpn <name> create <id-value> [<attribute>=<value> ...]
vpn <name> delete
vpn list
vpn listnames
vpn <name> show

vpn <name> get <attribute>
vpn <name> set <attribute>=<value> [<attribute>=<value> ...]
vpn <name> unset <attribute>
```

### Description

The vpn commands manipulate VPN objects in the DHCP server configuration. Each VPN must have a unique ID value. Other objects are associated with (or placed into) a VPN by VPN id.

Note: The reserved names "global" and "all" cannot be used as VPN names. These reserved names are used by other commands that take vpn names as arguments.

In contrast to most other CLI configuration objects, you can change the name of a VPN. The ID of the VPN, however, cannot be changed. Both the name and ID of the VPN must be unique upon creation.

### Examples

```
nrcmd> vpn red show

nrcmd> vpn red set vpn-id=23:456
```

### Status

### See Also

### Attributes

**addr-blocks-default-selection-tags** *string*

Specifies the default selection tag (or list of tags) that will be associated with incoming subnet-allocation requests in the VPN that do not contain any subnet name data. No default.

**addr-blocks-use-client-affinity** *bool*

Determines whether the DHCP server attempts to allocate subnets to clients using address-blocks that the clients have already used. Default is true (enable). If you disable this attribute, the server then supplies subnets from any block that is suitable, based on other selection data in the clients' messages.

**addr-blocks-use-lan-segments** *bool*

Controls whether DHCP subnet-allocation uses the lan-segment attribute when configured on address-blocks.

**addr-blocks-use-selection-tags** *bool*

Controls whether the server compares the subnet name data in incoming subnet-allocation requests with each address-block's selection tags. A block will only be considered if the two match. No default.

**description** *string*

Describes the VPN that this object represents.

**id** *int* required,unique,immutable

The VPN's unique id. This is a 32 bit value that is associated with every VPN-qualified IP address, subnet or address block. It is different than and unrelated to the standard 7-byte VPN id.

**name** *string* required,unique

The VPN's name within the CNR management system. Independent but could be the same as the vrf-name.

**vpn-id** *vpnid*

The vpn-id in RFC 2685 format (i.e., 7 octets), using a syntax similar to that used by IOS to enter the same information. The syntax is 3 hex octets, a colon, and 4 hex octets. For example 010203:04050607 would be the way to enter the following hex octets: 01:02:03:04:05:06:07 into this property.

**vrf-name** *string*

The VPN's VRF name.

## zone

zone - configures a DNS zone

### Synopsis

```
zone <name> create primary file=<hostfile> [template=<template-name>]
zone <name> create primary <name server> <person>
                        [template=<template-name>] [<attribute>=<value>...]
zone <name> create secondary <address> [<attribute>=<value>...]
zone <name> delete
zone list
zone listnames
zone <name> set =<value> [<attribute>=<value> ...]
zone <name> get <attribute>
zone <name> unset <attribute>
zone <name> disable <attribute>
```

```

zone <name> enable <attribute>
zone <name> show

zone <name> addHost <host name> <address> [<alias> ...]
zone <name> removeHost <host name>
zone <name> listHosts

zone <name> addRR [-staged|-sync] <name> [<ttl>] [<class>] <type> <data>
zone <name> addDNSRR <name> [<ttl>] <type> <data>
zone <name> removeRR <name> [<type> [<data>]]
zone <name> removeDNSRR <name> [<type>] [<data>]
zone <name> listRR [all|ccm|dns]
zone <name> findRR [-namePrefix <namePrefix>]
    [-rrTypes <rrTypeList>] [-protected | -unprotected]

zone <name> < protect-name|unprotect-name > <name>
zone <name> forceXfer secondary
zone <name> syncToDns
zone <secondary-zone-name> promote-to-primary

zone <name> chkpt
zone <name> dumpchkpt

zone <name> scavenge

zone <name> getScavengeStartTime

zone <name> applyTemplate <template-name>

```

## Description

The zone command lets you create and edit DNS zones.

The name of the zone may be an IPv4 subnet (<address>/<length>), IPv6 prefix (<address>/<length>), prefix name (the prefix address is used), or DNS name.

```

zone <name> addHost <host name> <address> [<alias> ...]
zone <name> removeHost <host name>
zone <name> listHosts

```

The addHost command adds a host with a given name, address and optional aliases to the zone.

The removeHost command removes a host from the zone.

The listHosts command lists the hosts in the zone.

```

zone <name> addRR [-staged|-sync] <name> [<ttl>] [<class>] <type> <data>
zone <name> addDNSRR <name> [<ttl>] <type> <data>
zone <name> removeRR <name> [<type> [<data>]]
zone <name> removeDNSRR <name> [<type>] [<data>]
zone <name> listRR [all|ccm|dns]
zone <name> findRR [-namePrefix <namePrefix>]
    [-rrTypes <rrTypeList>] [-protected | -unprotected]

```

The addRR command adds a protected resource record to a zone. The arguments to addRR are in the same format as BIND files. An attempt to add a protected record to an unprotected name will fail.

The `removeRR` command removes all specified protected resource records. Resource records may be specified by name, by name and type, or by name, type, and data (the data is specified in BIND-style format.)

The `addDNSRR` command adds an unprotected resource record. The name, type, and data must be specified. An attempt to add an unprotected record to a protected name will fail.

The `removeDNSRR` command removes all specified unprotected resource records. Resource records may be specified by name, by name+type, or name+type+data. The changes take effect immediately; no `serverreload` is necessary. If the DNS server is not running, the command will fail.

The `listRR` command lists the resource records in the zone. CCM records are the records being managed by the CCM server, and stored in its database. DNS records are the records that the running DNS server is serving to clients.

The `findRR` command displays the resource records matching a name prefix, a list of resource record types, and whether protected or not (or either).

The `cleanRR` command removes obsolete resource records. It is particularly useful for removing records remaining from zone deletion followed by recreation of the same zone. This command is valid only for pre-6.2 clusters.

**zone <name> protect-name|unprotect-name> <name>**

The `protect-name/unprotect-name` command sets the protection status of the resource records for the name. Protected names cannot be updated using DNS update requests.

**zone <name> forceXfer secondary**

The `forceXfer` command forces a full zone transfer of a secondary zone, regardless of the zone serial number, to synchronize DNS data store. If a normal zone transfer is already in progress, the `forceXfer` command is scheduled immediately after the normal zone transfer finishes. An option for primary zones has not been implemented yet.

**zone <name> chkpt**

**zone <name> dumpchkpt**

The `chkpt` command forces the specified zone name to be the next one checkpointed. If none are currently being checkpointed it is done immediately. Otherwise, it is done upon completion of the current checkpoint.

The `dumpchkpt` command interprets an existing checkpoint file and writes its contents to a file in human-readable format.

**zone <name> scavenge**

**zone <name> getScavengeStartTime**

The `scavenge` command schedules zone scavenging immediately for the given zone regardless of the scavenging interval.

The `getScavengeStartTime` command returns the date and time

of the next check for stale records, when records might be scavenged.

**zone** <secondary-zone-name> **promote-to-primary**

The `promote-to-primary` command can be used to promote a secondary zone to a primary zone (for example, if the primary DNS server has had a hardware failure).

**zone** <name> **applyTemplate** <template-name>

The `applyTemplate` command applies the specified zone template to the zone. All properties configured on the zone template are applied to the zone.

## Examples

```
nrcmd> zone example.com. create primary file=host.local
nrcmd> zone example.com. create primary ns ns-server
```

## Status

## See Also

[zone-template](#)

## Attributes

**checkpoint-interval** [rangetime](#)(60m-1w) default = 3h

Sets the number of seconds that elapse between saves of zone data. When the interval expires, Network Registrar takes a snapshot of the zone data and records it in the zone checkpoint data.

**checkpoint-min-interval** [rangetime](#)(60s-45m)

Specifies the minimum amount of time required (in seconds) between the time the first checkpoint occurs and the second checkpoint starts. This attribute applies only to zones with dynamic resource records.

**defttl** [rangetime](#)(0-68y5w3h14m7s) default = 24h, required

Controls the default TTL value used for resource records in a zone that do not specify a TTL.

**dist-map** [objref](#)(CCMZoneDistribution)

Identifies the zone distribution map associated with the specified zone. The zone distribution map describes which primary and secondary DNS servers should provide DNS service for this zone.

**dynamic** [bool](#) default = true

Enables RFC 2136 dynamic updates to the zone. The most common of these updates is a DHCP server.

**expire** [rangetime](#)(1s-68y5w3h14m7s) default = 1w

Sets the number of seconds that a secondary server can continue providing zone data without confirming that the data remains current. The expire interval must be greater than the refresh interval.

**minttl** [rangetime](#)(0-68y5w3h14m7s) default = 10m

Sets the minimum TTL value displayed in resource records in a zone. Records with TTL values lower than the `minttl` are rounded up to this value.

**nameservers** [dname](#) required

Lists the nameservers for this zone.

**notify** [bool](#)

Enables notification of other authoritative servers when zone changes. When set, to either true or false, it overrides the global "notify" value for this zone.

**notify-set** [ipaddr](#)

Lists additional servers to notify of changes to this zone, in addition to the servers listed in NS records for the zone, with the exception of the server described by the "ns" property of the zone (the "ns" field of the SOA record), receive notifications. Servers listed in "notify-list" are also notified.

**ns** [dname](#) required

Displays the fully-qualified domain name of the primary nameserver for this zone. This host is the original, or primary source, of zone data for this zone.

**nsttl** [dnsttl](#)

Displays the ttl value applied to the NS resource record for the zone.

**origin** [dname](#) required,immutable

Displays the fully-qualified name of the zone's root. The origin is the root of the zone.

**owner** [objref](#)(CCMOwner)

Names the owner of this zone. Use the owner field to group similarly owned zones and to limit administrative access.

**person** [dname](#) required

Displays a domain name specifying the mailbox of the person responsible for this zone. The first label is a user or alias, the rest of the labels are a mail destination. A mailbox of [hostmaster@test.com](#) would be represented as `hostmaster@test.com`.

**refresh** [rangetime](#)(1s-68y5w3h14m7s) default = 3h

Sets the number of seconds that a secondary server waits between polling for zone changes and refreshing its zone data.

**region** [objref](#)(CCMRegion)

Associates a region with the specified object. Use the region field to group similarly located zones and to limit administrative access.

**restrict-query-acl** [amelist](#)

Specifies the zone access control list (ACL) used to restrict the queries that the DNS server for this zone accepts. The ACL can contain host IPs, network addresses, TSIG keys, and ACLs. Only queries from clients defined in the ACL are accepted. If unset, the zone inherits the value of the server's restrict-query-acl attribute.

**restrict-xfer** [bool](#) default = false

Limits sending zone transfers to a specific set of hosts. To restrict zone transfers, use the restrict-xfer-acl attribute to specify the access control list that defines which servers can perform zone transfers.

**restrict-xfer-acl** [amelist](#)

Identifies the access control list designating who can receive zone transfers from this zone.

**retry** [rangetime](#)(1s-68y5w3h14m7s) default = 60m

Sets the number of seconds that a secondary server waits before it retries polling for changes to zone data or it retrieves zone data.

transfer that has encountered errors. The retry interval less than (expire - refresh).

**scvg-enabled** [bool](#) default = false

Enables dynamic resource-record scavenging for the zone. attribute removes stale records when clients are configured to perform DNS updates but do not delete their entries when they're no longer valid. If the DHCP server to perform updates, it will also delete records when client leases expire. Scavenging should not be enabled on these

**scvg-ignore-restart-interval** [rangetime](#)(2h-24h)

Ensures that the server does not reset the scavenging timer whenever a server restarts. With this attribute set, Network Registrar ignores the time between when a server goes down and the time it restarts. This interval is normally 2h. The value can range from two hours to one day. With any time longer than the set time, Network Registrar recalculates the scavenging period to allow for record updates that cannot take place while the server is stopped. You can set this attribute on a zone, and the value set on the zone overrides the server setting. Default is 2h.

**scvg-interval** [rangetime](#)(60m-1y)

Sets the period of time that must elapse before a DNS server can remove an out-of-date address (A) record. An A record becomes out-of-date once it ages past its initial creation date plus scvg-refresh-interval and scvg-no-refresh-interval. Default is 1w.

**scvg-max-records** [rangeint](#)(1-10000)

Sets the maximum number of records to remove from a zone during its scavenging interval.

**scvg-max-records-searched** [int](#)

Sets the maximum number of records to search for out-of-date A records. These records are candidates for scavenging.

**scvg-no-refresh-interval** [rangetime](#)(60m-1y)

With scavenging enabled, sets the interval during which DNS updates cannot increment an A record timestamp. After both the no-refresh and refresh intervals expire, the record is a candidate for scavenging. The value can range from one hour to 365 days. You can also set this attribute on a zone, and the value set on the zone overrides the server setting. Default is 1w.

**scvg-pause-interval** [rangetime](#)(1s-24h)

Sets the number of seconds the server waits after scavenging one set of records before going to the next set.

**scvg-refresh-interval** [rangetime](#)(60m-1y)

With scavenging enabled, sets the interval during which DNS updates can increment the A record timestamp. After both the no-refresh and refresh intervals expire, the record is a candidate for scavenging. The value can range from one hour to 365 days. You can also set this attribute on a zone, and the value set on the zone overrides the server setting. Default is 1w.

**serial** [int](#) required

Displays an administratively specified serial number. The number value must always increase; therefore, this serial number is only applied to the zone if it is greater than the actual serial number.

**soatl** [dnsttl](#)

Displays the time-to-live (ttl) value applied to the SOA resource record of the zone.

**subzone-forward** [enumint](#)(normal=0, no-forward=1) default = normal

Specifies whether subzones use forwarders or not. When not set, any query for the zone is not sent to the forwarder is an extended resolution exception.

**update-acl** [amelist](#)

Specifies the access control list for DNS updates to the given as an address match element list. The access control is not applied to administrative edits managed through the

**update-policy-list** [string](#)

Lists the DNS update policies used to authorize or deny updates. This attribute is ignored if the update-acl attribute is also set.

## zone-dist

zone-dist - Configures zone distributions

### Synopsis

```
zone-dist <name> create <primary-cluster> [<attribute>=<value> ...]
zone-dist <name> delete
zone-dist list
zone-dist listnames
zone-dist <name> show

zone-dist <name> get <attribute>
zone-dist <name> set <attribute>=<value> [<attribute>=<value> ...]
zone-dist <name> unset <attribute>

zone-dist <name> addSecondary <secondary-cluster>
    [<master-server-ipkeys>]
zone-dist <name> removeSecondary <secondary-cluster>
    [<master-server-ipkeys>]
zone-dist <name> listSecondaries

zone-dist <name> sync < update | complete | exact >
    [<no-rrs>] [<no-secondaries>]
```

### Description

The zone-dist command lets you define and manage zone distribution configurations.

On local clusters, the zone-dist sync command synchronizes staged edits to the DNS server and synchronizes primary zones to secondaries. Regardless of the mode selected, the exact list of authoritative zones (primary and secondary) is synchronized with the DNS server.

On the regional cluster, the zone-dist sync command synchronizes primary zones from the regional configuration to the primary local cluster, and synchronizes primary zones to secondaries. Primary zones on the local cluster are replaced in Update or Complete mode. In Exact mode, extra primary zones found on the local cluster are deleted.

Secondary servers use the same synchronization logic at both local and regional clusters. In Update mode, synchronization

ensures only that corresponding secondary zones exist on the server. In Complete mode, any existing zones are updated to use the master servers list specified by the distribution map. In Exact mode, any zones not matching the distribution map are deleted.

Use the [no-rrs] and [no-secondaries] flags to skip portions of the synchronization logic. While using these flags might improve the performance of the command, use them only when you are certain there are no changes pending. For example, if primary zones are current with the DNS server, you can use the [no-rrs] flag to synchronize your secondary zones.

On local clusters, RRs are always synchronized to the local DNS server if changes are pending, and thus the [no-rrs] flag is ignored.

## Examples

## Status

## See Also

[cluster](#)

## Attributes

**ixfr** [enumint](#)(enable=1, disable=2, use-server-settings=3) default = use-s

Configures secondary zones to enable incremental requests. When set, this attribute overrides the global ixfr-enable value. Using the server global (not setting this value per-zone) enables you to turn incremental transfers on or off or to set a policy for your zones and specific exceptions to global value.

- 1 enable  
Permits incremental transfers for this zone.
- 2 disable  
Prohibits incremental transfers for this zone.
- 3 use-server-settings  
Uses the server settings. Unset the value for all secondary zones.

If you set incremental transfers, then this zone operates differently than zones inheriting the server global value. If you unset incremental transfers, Network Registrar uses the server global ixfr value (and the operation of this feature specifically depends on this, since it has no way to override this per-zone feature).

**master-servers** [ipkey](#)

Lists the master-servers to set when creating secondary zone on secondary server.  
Validation: A list of one or more ip addresses with key name (format [-]).

**name** [string](#) required,unique

Names this zone distribution map.

**notify** [enumint](#)(enable=1, disable=2, use-server-settings=3, use-primary

## **default = use-server-settings**

Configures secondary zones to notify other authorities when there are zone changes.

- 1 enable  
Allows notification of other authorities when this zone changes. Overrides the global notify setting for this zone.
- 2 disable  
Disallows notification of other authorities when this zone changes. Overrides the global notify setting for this zone.
- 3 use-server-settings  
Uses the server setting. Unset the value for secondary zones.
- 4 use-primary-zone-settings  
Uses the value set at the primary zone to configure secondary zone.

## **notify-set ipaddr**

Provides an optional list of servers to notify when there are zone changes. Use with the notify attribute to configure secondary zones. All servers listed in NS records for the zone, with the exception of the server described by the mname field of the SOA record, are notified. Servers listed in the notify-set attribute are also notified.

## **primary oid(CCMCluster)**

Identifies the cluster or HA DNS pair serving the zone associated with this zone distribution map.

## **restrict-query enumint(use-map-settings=1, use-server-settings=2, use-primary-zone-settings=3) default = use-primary-zone-settings**

Determines how to restrict zone queries.

- 1 use-map-settings  
Uses the restrict-query-acl attribute to configure secondary zones.
- 2 use-server-settings  
Uses the server settings. Unset the value for secondary zones.
- 3 use-primary-zone-settings  
Uses the primary zone value to configure secondary zones.

## **restrict-query-acl amelist**

Configures the ACL that the zone uses to restrict zone queries. The DNS server honors the restriction set on the restrict-query attribute if it is set to use-map-settings. You can configure secondary zones. You can configure to include any of the following:

- host IPs
- network addresses
- TSIG keys
- global ACLs

Those clients defined in this ACL list are given access. If missing, the zone inherits the value of the server's restrict-query-acl attribute.

## **restrict-xfer enumint(enable=1, disable=2, use-server-settings=3, use-primary-zone-settings=4) default = use-primary-zone-settings**

Determines what settings to use to restrict zone transfers.

- 1 enable

- Limits zone transfers to a specific set of IP addresses. If you limit zone transfer, you must then restrict-xfer-acl attribute to list the servers allowed to perform zone transfers.
- 2 `disable`  
Places no restrictions on zone transfers.
- 3 `use-server-settings`  
Use the server settings and unsets the value for secondary zones.
- 4 `use-primary-zone-settings`  
Configures a secondary zone with the primary zone's value.

### **restrict-xfer-acl** [amelist](#)

Configures the access control list that designates which servers can receive zone transfers from the specified zone. If the restrict-xfer attribute is set to `use-server-settings`, the

## zone-template

`zone-template` - Configures a zone template

### Synopsis

```

zone-template <name> create [<attribute>=<value> ...]
zone-template <name> delete
zone-template <name> set <attribute>=<value> [<attribute>=<value> ...]
zone-template <name> get <attribute>
zone-template <name> unset <attribute>

zone-template <name> disable <attribute>
zone-template <name> enable <attribute>
zone-template <name> show

zone-template <name> create clone=<clone-name>
zone-template <name> apply-to [all | <zone1>[,...]]

```

### Description

The `zone-template` command lets you configure templates to use when creating zones.

### Examples

### Status

### See Also

[zone](#)

### Attributes

#### **checkpoint-interval** [rangetime](#)(60m-1w) default = 3h

Sets the number of seconds that elapse between saves of zone data. When the interval expires, Network Registrar takes a snapshot of the zone data and records it in the zone checkpoint data file.

#### **checkpoint-min-interval** [rangetime](#)(60s-45m)

Controls the minimum time required (in seconds) between time the first zone checkpoint occurs and the second zone checkpoint occurs.

checkpoint starts. This attribute applies only to zone dynamic rrs.

**defttl** **rangetime**(0-68y5w3h14m7s) default = 24h, required

Controls the default TTL value used for resource records in a zone that do not specify a TTL.

**dist-map** **oid**

Associates a zone distribution map with a zone. The map describes the primary and secondary DNS servers that provide DNS service for this zone.

**dynamic** **bool**

Enables RFC 2136 dynamic updates to the zone. The most common of these updates is a DHCP server.

**expire** **rangetime**(1s-68y5w3h14m7s) default = 1w

Sets the number of seconds that a secondary server can provide zone data without confirming that the data is current. The expire interval must be greater than the refresh interval.

**minttl** **rangetime**(0-68y5w3h14m7s) default = 10m

Sets the minimum TTL value displayed in resource records in a zone. Records with TTL values lower than the minttl are updated with this value.

**name** **string** required,unique

Names this zone template.

**nameservers** **string**

Lists the nameservers for a zone.

**notify** **bool**

Enables notification of other authoritative servers when the zone changes. When set, to either true or false, it overrides the global "notify" value for a zone.

**notify-set** **ipaddr**

Lists additional servers to notify of changes to a zone, in addition to the servers listed in NS records for the zone, with the exception of the server described by the "ns" property of the zone (the field of the SOA record), receive notifications. Servers listed in "notify-list" are also notified.

**ns** **string**

Specifies the fully-qualified domain name (FQDN) of the primary name server for the zone. This host is the original or primary source of data for this zone.

Note: Network Registrar treats the value set here as a host name rather than a dnsname to enable encoding relative or absolute names in this attribute.

**nsttl** **dnsttl**

Controls the ttl value applied to the zone's NS resource records.

**owner** **objref**(CCMOwner)

Identifies the owner of this zone. Use the owner field of the zone to identify similarly owned zones and to limit administrative access.

**person** **string**

Specifies the mailbox for the hostmaster (person) in domain form. The first label is a user or mail alias, the rest of the labels are a mail destination. A mailbox of [hostmaster](#) would be represented as:

hostmaster.test.com.

Note: Network Registrar treats the value set here as a rather than a dnsname to enable encoding relative or absolute names in this attribute.

**refresh** `rangetime(1s-68y5w3h14m7s)` default = 3h

Sets the interval at which a secondary server contacts master server for changes to zone data. The interval is defined in the server's SOA record and is also known as secondary refresh time.

**region** `objref(CCMRegion)`

The region associated with this object. This region filters to group similarly located zones and can be used to limit administrative access.

**restrict-query-acl** `amelist`

Zone Address Control List (ACL) used to restrict queries honored by the DNS server on this zone. This list can contain IPs, network addresses, TSIG keys and/or (global) ACLs. Only clients defined in this ACL list shall be honored; others are refused. If unset, the zone inherits the value of the server's restrict-query-acl attribute.

**restrict-xfer** `bool`

Restricts sending zone transfers to a specific set of hosts. If restrict zone transfers, you need to use the restrict-transfer-acl property to specify the access control list of who is allowed to perform zone transfers.

**restrict-xfer-acl** `amelist`

The access control list that designates who is allowed to perform zone transfers from this zone.

**retry** `rangetime(1s-68y5w3h14m7s)` default = 60m

Sets the amount of time that a secondary server waits before it retries polling for changes to zone data or it retries a zone transfer that has encountered errors. The retry interval is less than the expire and refresh intervals. A good value is between one-third and one-tenth of the refresh time.

**scvg-enabled** `bool`

Enables dynamic resource-record scavenging for the zone. When this feature is enabled, the server will remove stale records that arise when clients are configured to perform DNS updates, but do not delete their records when they are no longer valid. If the DHCP server performs updates, it also deletes records when client leases expire. Scavenging should not be enabled on these zones.

**scvg-ignore-restart-interval** `rangetime(2h-24h)`

Ensures that the server does not reset the scavenging period whenever a server restarts. Within this attribute set, Network Registrar ignores the time between when a server goes down and the time it restarts. This interval is normally set to 24 hours. The value can range from two hours to one day.

With any time longer than the set time, Network Registrar recalculates the scavenging period to allow for record updates that cannot take place while the server is stopped. You can set this attribute on a zone, and the value set on the server overrides the server setting.

**scvg-interval** `rangetime(60m-1y)`

Sets the seconds that DNS waits before removing (scavenging) out-of-date resource records.

**scvg-max-records** `rangeint(1-10000)`

Controls the maximum number of records to search at one candidates to be scavenged.

**scvg-max-records-searched** [int](#)

Sets the maximum number of records to be scavenged for during a scavenging interval.

**scvg-no-refresh-interval** [rangetime](#)(60m-1y)

Sets the number of seconds during which DNS updates can the zone timestamp.

**scvg-pause-interval** [rangetime](#)(1s-24h)

Controls the interval (in seconds) that scavenging will after scavenging a set of records, before going onto the set.

**scvg-refresh-interval** [rangetime](#)(60m-1y)

Sets the number of seconds during which DNS updates can the zone timestamp. After both the no-refresh and refresh expire, the record is a candidate for scavenging. The value can range from one hour to 365 days. The zone setting (the server setting of 604800s (1w)).

**serial** [int](#)

Sets the starting serial number of the zone. A DNS server a serial number to indicate database changes. Increment number trigger zone transfers to a secondary server.

**soattl** [dnsttl](#)

Controls the ttl value applied to a zone's SOA resource

**subzone-forward** [enumint](#)(normal=0, no-forward=1) default = normal

Specifies whether subzones use forwarders or not. When set, any query for the zone will not be sent to the forwarder is an extended resolution exception.

**update-acl** [amelist](#)

Specifies the access control list for DNS updates to a defined as an address match element list. The access control list is not applied to dynamic updates coming from the from them UIs are always allowed as long as the zone at dynamic is enabled. The access control list is not applied to administrative edits managed through the CCM server.

**update-policy-list** [string](#)

An ordered list of DNS update policies that can be used or deny DNS updates. This attribute will be ignored if is also set.

---

## Using the *nrcmd* Program in Scripts

You can use the *nrcmd* command to interactively configure and control a Cisco Network Registrar cluster, or you can use it as a programming interface for another program or script.

### Connecting to Network Registrar

When you use the *nrcmd* command, you connect to a Network Registrar *cluster* to read and write configuration data, read state data, and perform control operations.

A Network Registrar cluster consists of:

- The data manager, the MCD server, which controls access to persistent datastores that contain configuration and state information for the DNS, DHCP, and TFTP servers.
- The server agent, AIC Server Agent, which starts and stops the protocol servers, and provides a standard control interface to them.
- The DNS, DHCP, and TFTP protocol servers.

### Performing Authentication

When you log in to a cluster you are authenticated with a name and a password. The authentication protocol uses one-way hashes so that a password does not travel across the wire. In interactive mode, the *nrcmd* command prompts for a valid username and password. You can also provide the username or password on the command line, in the environment, or in the Windows Registry. (On Solaris, the Windows Registry is emulated by files in the product configuration directories.)

Because you may not want to embed the administrator password in a command script, the environment variables and registry entries provide alternate locations with different visibility levels. The environment variables `AIC_CLUSTER`, `AIC_NAME`, and `AIC_PASSWORD` specify the cluster, administrator name and administrator password values, respectively. These are similar to the same registry keys in the directory `HKEY_CURRENT_USER\Software\American Internet\Network Registrar\2.0`.

### Choosing Scripting Techniques

Because *nrcmd* does a significant amount of processing at connect time, it is more efficient to perform multiple commands in a single session rather than to initiate a distinct connection and login for each command. The simplest way to have a single *nrcmd* session perform multiple commands is to create a batch file with one command per line and to redirect standard input from that file. A more complicated approach, but one that provides more control over the command sequence, is to run *nrcmd* from a controlling program and have that program send commands and read their status and output.

### Using *nrcmd* Batch Files

The simplest way to automatically perform multiple configuration commands is to create a batch file of *nrcmd*

commands and have them executed sequentially. For example, to create a scope and add reservations to it, you can enter these commands and store them in the file `scope.txt`. Lines beginning with the pound character (`#`) are comment lines:

```
# This set of commands creates a scope and adds four reservations

scope demo1 create 24.10.2.0 255.255.255.0

scope demo1 addReservation 24.10.2.1 1,6,0a:23:45:67:89:01

scope demo1 addReservation 24.10.2.2 1,6,0c:23:45:67:89:02

scope demo1 addReservation 24.10.2.3 1,6,0c:23:45:67:89:03

scope demo1 addReservation 24.10.2.4 1,6,0a:23:45:67:89:04
```

Note: End the last command line with a newline character, or the command will not be executed.

You can then run a single `nrcmd` session to execute all of these commands.

```
% nrcmd -b < scope.txt
```

The advantage to using batch files is that you can execute multiple configuration commands while only incurring the connection cost once. However, if a command fails (such as the initial scope creation in the previous example), the batch file continues even though subsequent commands are now useless.

You can use the `assert` function of the `session` command to perform simple logic checks. This command allows a `nrcmd` batch script to assert that a given condition is true. If the condition is true, the command has no effect; if false, the batch file is terminated at that point. For example, before executing the set of `scope` commands in the `scope.txt` file in the previous example, you might want to ensure that the cluster is locked.

```
# Quit if cluster cannot be locked

session assert locked
```

## Command Syntax

When you execute `nrcmd` commands that contain equal-signs, you must put them within quotation marks. For example, to use a single command to create a client-class name, enter:

```
nrcmd -C cluster -N name -P password "client MAC create client-class-name= name "
```

## Adding Program Control

A more sophisticated method for automatically configuring and controlling Network Registrar is to have a program or script start a `nrcmd` session and communicate with the session through standard input and output.

To control `nrcmd` from another program, you need to start `nrcmd` from the controlling program and redirect standard input and output from `nrcmd` to file handles in the controlling program. The controlling program can then write commands to the input file handle and read results from the output file handle.

When running in batch mode, *nrcmd* reads a line of input at a time and prints a new line after the prompt. This provides an easily parsed sequence of lines in response to any command where:

- The syntax is *status-line result-lines prompt-line*
- The *status-line* has the format `[0-9]{3} .*`.
- There may be zero or more *result-lines* of any format.
- The *prompt-line* is *nrcmd>* .

The exact details of starting up *nrcmd* as a child process, and writing to and reading from its standard input and output, are specific to the programming language you use to implement the controlling program.

## CLI Codes and Formats

[Status Returns](#)

[Network Registrar Error Codes](#)

[Import and Export File Formats](#)

### Status Returns

The **nrcmd** program returns status information on the first line of information written to the standard output stream. If there is more data, **nrcmd** displays this information on additional lines.

The first line consists of a numeric status that is followed by a human-readable error status.

The status codes are all three-digit integer decimal numbers. The range 100 through 599 is grouped as in [Table 4-1](#).

**Table 4-1. Status Codes**

Table 4-1 Status Information

Value	Description
100-199	Normal return
200-299	Informational (warning)
300-499	Error
500-599	Fatal Error

For anything other than an error, Network Registrar assumes that the requested operation was completed; however, some warning messages signal a condition that must be corrected. Unless a fatal error occurs, the command line interface will keep running in interactive mode. Fatal errors imply that something serious happened and that you must restart the Network Registrar command line processor.

### Network Registrar Error Codes

[Table 4-2](#) lists and describes the Network Registrar error codes.

**Table 4-2. Error Codes**

Number	Description
--------	-------------

100	OK
101	OK, with warnings
102	Lease already reserved to this client
103	Lease not reserved
104	Lease deleted along with reservation
105	Lease created along with reservation
106	Assertion failed
107	VPNID currently in use by at least one scope
204	Assigned host is not contained in pool zone
300	Unexpected error
301	No server found
302	Not found
303	Read only property
304	No policy found
305	Too many
306	Unknown command
307	Unknown keyword
308	Unknown parameter
309	Too many arguments
310	Too few arguments
311	No response to lease request
312	Unexpected response from server
313	No match
314	Duplicate object
315	Import failure
316	Invalid
317	Open failed
318	No MAC address found
319	No lease found
320	Generic error
321	Invalid name
322	Feature not supported
323	Read error
324	Invalid IP address list
325	Invalid type
326	ODBC database access error
327	IP address not contained within pool subnet
328	Identical MX resource record already exists
329	Identical TXT resource record already exists
330	Address is not contained in pool
331	Host is already assigned to an address
332	Address is already assigned to a host
333	No unassigned IP address found in pool

334	Address has not been assigned to a host
335	Static address pools are not enabled, create a pool to enable
336	Range overlaps another pool
337	Host has multiple IP address assignments
338	Address has multiple host assignments, must supply <name> argument
339	Expected unsigned 16-bit preference value
340	ODBC 3.x or higher required
350	Generic DHCP error
351	Cannot resolve partner name to an IP address
352	No failover object for specified partner
353	Still communicating with partner
354	Server is already in partner-down state
355	Can't set partner-down while in recover
356	Not allowed in read-only mode
357	Not a secondary
358	Not a primary
359	No zone matched
360	Forcexfer for this zone is already scheduled
361	Lease is not reserved
362	Scope unknown in server
363	Invalid IP address in server
364	Invalid MAC address in server
365	Failure creating MAC address in server
366	Unknown object in server
367	Command not supported by server
368	Bad length in server
369	Inconsistent scope in server
370	updateSMS not configured in server
371	Server out of memory
372	SMS interface .dll did not load correctly
373	updateSMS already processing in server
374	Invalid word array
375	updateSMS invalid argument
376	Lease is reserved to a different client
377	Client already reserved a different lease
378	Field name or number not found
379	Suboption name or number already exists
380	Suboption name or number not found
381	Invalid character '-' in vendor-option name
382	Data not found for vendor-option
383	Field name or number already exists
384	counted-array can only be used with array types
385	Read-only attribute cannot be modified

386	Required attribute cannot be unset
387	Invalid vpn-id
388	Invalid IP address syntax
389	Invalid vpn name
390	Invalid IP address value
391	Invalid vpn specification
392	Cannot remove session's current vpn
393	Duplicate property
394	Invalid vpn-id specification
395	Lease has no scope pointer -- internal error
396	Invalid vpn-id specification
397	Expression too long
398	Insufficient memory
401	Login failure
402	Permission denied
403	Couldn't lock database
404	Login required
405	Invalid license key
406	A lock is required for this operation
407	Unable to release lock
408	Unable to obtain lock
409	Couldn't lock static address pool
501	Connection failure
502	Server failure
503	Cluster version failure
504	Wrong license key type, local cluster key type needed
504	Wrong license key type, regional key type needed
505	Connected to wrong cluster type
506	Duplicate option id
507	Duplicate option name
508	Duplicate vendor-option-enterprise-id
509	Duplicate vendor-option-string
510	Version mismatch
511	No version found
512	Incomplete object
513	Incomplete attribute
514	Incomplete NLIST
515	Invalid repeat count

## Import and Export File Formats

This section describes the import leases and export leases file format.

The syntax is:

field1|field2|field3|...

The fields are listed next. If, in the import file, you chose not to supply the information for an optional field, you need to use delimiters ( | ) so that the number of fields is still 12. For example, type xyz|abc||123:

MAC address in xx:xx:xx:....:xx format (required)

MAC address type (required)

MAC address length (required)

IP address in dotted decimal format, a.b.c.d (required)

Start of lease time (GMT) (optional)

Lease expiration time (GMT) (optional)

Allowable extension time (GMT) (optional)

Last transaction time (GMT) (optional)

IP address of the DHCP server (optional)

Host name (without domain) (optional)

width=19 border=0>Domain name (optional)

Client ID (optional)

VPN (optional)

---

**Note:** For all the time fields, you can use either the number of seconds since 1970, or day, month, date, time, year format (Mon Apr 13 16:35:48 1998).

---

All contents are Copyright © 1992--2007 Cisco Systems, Inc. All rights reserved.

Angle brackets (< >)—Indicate you must provide a value for an attribute or argument in the command.

Copyright © 1992--2007 Cisco Systems, Inc. All rights reserved.

---



# Network Registrar Attribute Types

## **AT\_AMELST** (amelist)

A list of AddressMatchElement objects; structurally equivalent to [AT\\_NLIST](#) (AddressMatchElement). This specific type is provided to make generic parsing and unparsing work for attributes of this type.

## **AT\_ARRAY** (array)

A sequence of any other type. This attribute type is deprecated in favor of the more general [AT\\_NLIST](#) type.

## **AT\_ATTRTYPE** (attrtype)

A CNR attribute type, stored as an integer.

## **AT\_BITARRAY** (bitarray)

A bit-array, stored as an AT\_BLOB of a fixed size.

## **AT\_BLOB** (blob)

A sequence of unsigned octets.

## **AT\_BOOL** (bool)

An 8-bit boolean value with only 0 and 1 as the legal values.

## **AT\_BYTE** (byte)

An unsigned 8 bit integer value.

## **AT\_CALCBIT** (calcbit)

A specialized type used to in representing certain DHCP options, such as option-81, the 'client-fqdn' option.

## **AT\_CALCFLAG** (calcflag)

A value whose type is defined by the byte value that precedes it. Example: dhcp-v4 option-122, suboption-3. The extra-value field contains a map between the flag-byte and the AT\_xxx type to use for parse/unparse.

## **AT\_CIS** (cis)

A string that has case-insensitive comparison properties.

## **AT\_CLEARTEXT** (clrtxt)

A string attribute that contains the clear text of sensitive information, such as a password.

## **AT\_DATE** (date)

A 32-bit integer value representing a point in time to 1 second resolution.

## **AT\_DICT** (dict)

Deprecated

## **AT\_DNSNAME** (dname)

A fully qualified DNS name, encoded in DNS wire format with counted labels.

**AT\_DNSTTL** (dnsttl)

A signed integer with the semantics of a DNS ttl -- only -1 is allowed as a negative number, with the special meaning 'use the zone default'.

**AT\_ENUMBYTE** (enumbyt)

An 8-bit integer with a fixed set of valid values. Each of the values can optionally have a string name associated with it.

**AT\_ENUMINT** (enumint)

A 32-bit integer with a fixed set of valid values. Each of the values can optionally have a string name associated with it.

**AT\_ENUMSHORT** (enumshort)

A 16-bit integer with a fixed set of valid values. Each of the values can optionally have a string name associated with it.

**AT\_ENUMSTR** (enumstr)

A string with a fixed set of valid values.

**AT\_ESTRING** (estr)

Deprecated.

**AT\_EXPR** (expr)

An embedded object of class Expression. This expression can be evaluated (in an appropriate context) to produce a typed value.

**AT\_FILESIZE** (filesize)

An unsigned 32-bit integer representing the size of a file.

**AT\_FILTER** (filter)

Deprecated.

**AT\_FLAGSINT** (flags)

A 32-bit integer with distinguished names associated with each bit position.

**AT\_GENADDR** (genaddr)

Deprecated.

**AT\_IFNAME** (ifname)

Deprecated.

**AT\_INT** (int)

An unsigned 32-bit integer.

**AT\_INT100** (int100)

An unsigned 32-bit integer number of 1/100's value. This is used for storing percentages in integer form.

**AT\_INT8** (decimal-byte)

A one-octet value with the same semantics as [AT\\_INT](#).

**AT\_INTI** (inti)

An unsigned 32-bit integer in Intel byte order.

**AT\_IP6ADDR** (ip6addr)

A 128-bit IPv6 address.

#### **AT\_IP6NET** (ip6net)

An IPv6 address and a one-octet prefix-length bit count.

#### **AT\_IPADDR** (ipaddr)

A 32-bit IPv4 address.

#### **AT\_IPKEY** (ipkey)

An IP address that can be associated with a port number and/or a required TSIG key name.

The canonical textual values are:

- [<address>](#)
- [<address>:<port>](#)
- [<address>:<port>-<key>](#)
- [<address>-<key>](#)

#### **AT\_IPNET** (net)

An IPv4 address and a count of the bits that comprise the network number.

#### **AT\_IPPAIR** (ippair)

A pair of IPv4 addresses, stored in 8 octets.

#### **AT\_KEY** (key)

A sequence of bytes holding a shared-secret key. These have a base-64 textual representation rather than the standard [AT\\_BLOB](#) representation.

#### **AT\_LISTREF** (listref)

Deprecated.

#### **AT\_MACADDR** (macaddr)

A MAC address, most frequently a 6 byte ethernet address with type 1, but more generally an arbitrary 1 byte type id, a 1 byte length and then 'length' address bytes.

#### **AT\_MASK** (mask)

An [AT\\_IPADDR](#) that is in the form of an IP address mask (its binary sequence matches 1\*0\*).

#### **AT\_MESSAGE** (dhcpmsg)

A DHCP message type.

#### **AT\_MSDATE** (msdate)

A 64-bit value that represents a specific point in time to millisecond resolution.

**AT\_MSG6** (msg6)

A DHCPv6 message.

**AT\_MSTIME** (mstime)

A 64-bit integer value that represents a span of time to millisecond resolution.

**AT\_NAMEREF** (nameref)

A string that refers to another object by name.

**AT\_NDICT** (ndict)

Deprecated.

**AT\_NLIST** (nlist)

A list of elements of some other type. This is the preferred type for storing lists.

**AT\_NODE** (dhcnode)

Deprecated.

**AT\_NOLEN** (no length)

A DHCP option code with no length or value. For example: PAD or END.

**AT\_NSTRING** (nstr)

A string that is stored as a counted sequence, and is not necessarily null-terminated.

**AT\_OBJ** (obj)

A CNR object of any schema class.

**AT\_OBJREF** (objref)

A reference to a specific class of object by OID. It is similar to an [AT\\_OID](#), but adds the additional expectation that the referenced object has the specified class and does exist in the database.

**AT\_OID** (oid)

An 8-byte object id. The AT\_OID type differs from the [AT\\_OBJREF](#) in that it does not imply that the OID can be resolved to any specific type of object.

**AT\_OPTION** (option)

A DHCPv4 option value.

**AT\_OPTION6** (option6)

A DHCPv6 option value.

**AT\_OPTIONID4** (optionid4)

A DHCPv4 option number, stored as a four-octet integer.

**AT\_OPTIONID6** (optionid6)

A DHCPv6 option number, stored as a four-octet integer.

**AT\_OVERLOAD** (overld)

A type representing the DHCP option-overload option as a single octet.

**AT\_PACK** (pack)

An [AT\\_BLOB](#) that has some kind of structure associated with it.

**AT\_PAD** (dhcppad)

The DHCP PAD option, a single zero octet.

**AT\_PCV** (pcv)

A 32 bit product compatibility version number. The components of this number are (from high to low): major: 8 bits, minor: 8 bits, revision: 16 bits.

**AT\_PERCENT** (percent)

An integer bounded to the range 0-100, with a normal ascii output form that ends with a '%'. The value is stored as a single byte.

**AT\_PREFIX** (prefix)

A 17-byte sequence representing an IPv6 prefix. It consists of 16 bytes of prefix (with bits beyond the prefix-length being 0) followed by 1-byte of prefix-length (0-128). This is like [AT\\_SUBNET](#) for IPv6

**AT\_RANGE** (range)

A 64-bit value containing a pair of 32-bit integers defining a range of integer values. This differs from the bounded integer and bounded time types.

**AT\_RANGEBYTE** (rangebyte)

An [AT\\_BYTE](#) value that is restricted to a range of valid values.

**AT\_RANGEINT** (rangeint)

An [AT\\_INT](#) value that is restricted to a range of valid values.

**AT\_RANGESHORT** (rangeshort)

An [AT\\_SHORT](#) value that is restricted to a range of valid values.

**AT\_RANGETIME** (rangetime)

An [AT\\_TIME](#) value that has an associated range of valid values.

**AT\_RDNSNAME** (rdname)

A relative DNS name, encoded in DNS wire format with counted labels.

**AT\_REQUEST** (dhcpreq)

A DHCP REQUESTED\_OPTIONS option, which the DHCP client uses to request option data by option number.

**AT\_RETCODE** (retcode)

A status code or return code.

**AT\_ROF\_FQDN** (rel or full fqdn)

A relative or fully qualified dns name.

**AT\_RULE** (rule)

A rule string, used in a DNS update-policy.

**AT\_SBYTE** (sbyte)

A signed 8 bit integer value.

**AT\_SECRET** (secret)

An object reference [AT\\_OID](#) to a Secret object that stores the actual secret.

**AT\_SET** (set)

Deprecated.

**AT\_SHORT** (short)

An unsigned 16-bit integer.

**AT\_SHORTPAIR** (shortpair)

A 32-bit value holding a pair of 16-bit unsigned integers.

**AT\_SHRTI** (shorti)

An unsigned 16 bit integer in Intel byte order.

**AT\_SINT** (sint)

A signed 32-bit integer.

**AT\_SINT8** (s-decimal-byte)

A one-octet value with the same semantics as [AT\\_SHORT](#).

**AT\_SINTI** (sinti)

A signed 32-bit integer in Intel byte order.

**AT\_SSHORT** (sshort)

A signed 16-bit integer.

**AT\_SSHRTI** (sshorti)

A signed 16 bit integer in Intel byte order.

**AT\_STIME** (stime)

A signed version of the [AT\\_TIME](#) type that allows negative time spans. The main use of this type is for time-zone offsets that may be positive or negative.

**AT\_STRING** (string)

A null-terminated sequence of ASCII bytes.

**AT\_STRUCT** (struct)

Deprecated.

**AT\_SUBNET** (subnet)

An IPv4 address and a count of the bits that comprise the network number. The address component will have its host bits set to 0. A similar type, [AT\\_IPNET](#) does not assume or require the host bits to be 0.

**AT\_TEXPR** (texpr)

A typed expression; its structure and representation is the same as an [AT\\_EXPR](#) but the extra value of the attribute contains the expected type of the value produced by evaluating this expression.

**AT\_TIME** (time)

An unsigned integer number of seconds. It differs from the [AT\\_DATE](#) type in that it encodes a span or duration of time rather than an exact point in time.

**AT\_TLV** (tlv)

A generic type-length-value tuple, in which the 'type' and 'length' are each one octet long.

**AT\_TLV2** (tlv2)

A generic type-length-value tuple, in which the 'type' and 'length' are each two

octets long.

#### **AT\_TLV4** (tlv4)

A generic type-length-value tuple, in which the 'type' and 'length' are each four octets long.

#### **AT\_TYPECNT** (type-cnt)

This is a special-purpose type used in configuring DHCP options used by PXE clients. It holds a repeating pattern of tuples, each consisting of a two-octet type, a length byte, and 'length' bytes of data.

#### **AT\_VENDOR\_CLASS** (vendor-class)

An [AT\\_BLOB](#) representing a DHCP vendor-class option. An enterprise ID is followed by opaque data. (If DHCPv4, the enterprise ID is followed by an EID length.)

#### **AT\_VENDOR\_NOLEN** (vendor-nolen)

An [AT\\_BLOB](#) representing a DHCP vendor-class option. An enterprise ID is followed by tuples of vendor-specific data. The enterprise ID is never followed by an ID-length.

#### **AT\_VENDOR\_OPTS** (vendor-opts)

An [AT\\_BLOB](#) representing DHCP vendor-specific options data. An enterprise-id that is followed by TLVs of vendor-specific data. If DHCPv4, the enterprise ID is followed by an EID length.

#### **AT\_VERSION** (version)

A two-component 32 bit number, with 16 bits of major version and 16 bits of minor version.

#### **AT\_VPNID** (vpnid)

A 7-byte standard VPN ID, as defined in RFC 2685.

#### **AT\_ZEROSIZE** (zero-size)

A sequence of bytes representing certain DHCP options that have an option code and a length, where the length is always zero.

---