



## CHAPTER 26

# Managing DHCPv6 Addresses

---

Cisco Network Registrar supports the following IPv6 addressing for DHCP (DHCPv6):

- **Stateless autoconfiguration (RFC 3736)**—The DHCPv6 server does not assign addresses, but instead provides configuration parameters, such as DNS server data, to clients.
- **Stateful autoconfiguration (RFC 3315)**—The DHCPv6 server assigns nontemporary or temporary addresses and provides configuration parameters to clients.
- **Prefix Delegation (RFC 3633)**—The DHCPv6 server delegates prefixes to clients (routers).

The DHCPv6 service provides these capabilities:

- **Links and prefixes**—Similar to DHCPv4 networks and scopes that define the network topology. Each link can have one or more prefixes.
- **Policies and options**—You can assign attributes and options to links, prefixes, and clients.
- **VPN support**—Provides multiple address spaces (virtual private networks).
- **Client-classing**—You can classify clients and select prefixes based on known clients or packet-based expressions.
- **Static reservations**—Clients can receive predetermined addresses.
- **Extensions**—Extend the DHCP server processing by using C/C++ and Tcl extensions.
- **DNS Updates**—DNS server updates of DHCP activity (over IPv4).
- **SNMP traps**—Generate traps for events, such as if the number of leases in a prefix exceeds a certain limit (or drops below a certain limit) or if the server detects duplicate addresses.
- **Statistics collection and logging**—Provides server activity monitoring.

The DHCPv6 service requires that the server operating system support IPv6 and that you configure at least one interface on the system for IPv6.

### See Also

[DHCPv6 Concepts, page 26-2](#)

[DHCPv6 Configuration, page 26-11](#)

[DNS Update for DHCPv6, page 26-33](#)

# DHCPv6 Concepts

The following subsections describe the concepts related to DHCPv6 operation:

- [IPv6 Addressing](#)
- [Links and Prefixes](#)
- [DHCPv6 Clients and Leases, page 26-5](#)
- [DHCPv6 Policy Hierarchy, page 26-9](#)
- [DHCPv6 Options, page 26-10](#)

## IPv6 Addressing

IPv6 addresses are 128 bits long and are represented as a series of 16-bit hexadecimal fields separated by colons (:). The A, B, C, D, E, and F in hexadecimal are case insensitive. For example:

```
2001:db8:0000:0000:0000:0000:0000:0000
```

A few shortcuts to this addressing are:

- Leading zeros in a field are optional, so that you can write 09c0 as 9c0, and 0000 as 0.
- You can represent successive fields of zeros (any number of them) by a double colon (::), but only once in an address (because, if used more than once, the address parser has no way of identifying the size of each block of zeros). This reduces the length of addresses; for example, 2001:db8:0000:0000:0000:0000:0000:0000 can be written:

```
2001:db8:::
```

Link-local addresses have a scope limited to the link, and use the prefix fe80::/10. Loopback addresses have the address ::1. Multicast addresses have the prefix ff00::/8 (there are no broadcast addresses in IPv6).

The IPv4-compatible addresses in IPv6 are the IPv4 decimal quad addresses prefixed by ::. For example, you can write the IPv4 address interpreted as ::c0a8:1e01 in the form ::192.168.30.1.

## Links and Prefixes

The explicit DHCPv6 configuration objects are links and prefixes:

- **Link**—Network segment that can have one or more prefixes, and adds an additional layer at which policies can be applied for DHCPv6 clients.
- **Prefix**—Equates to a scope in IPv4. The link associated with a prefix is similar to a primary scope, except that it names a link and not another prefix.

Just as with scopes, you can create multiple prefix objects for the same IPv6 prefix. However, rather than supporting multiple ranges with explicit start and end addresses, prefixes support only a single range that must be an IPv6 prefix with a length the same as, or longer than, the prefix object. For example, if you define a 2001::/64 prefix with a 2001::/96 range, the server can assign addresses from 2001:0:0:0:0:0:0:0 through 2001:0:0:0:0:0:0:ffff:ffff only. The range:

- Is limited to powers of 2.
- Must be unique (cannot be duplicated by any other range, except in a different VPN).
- Cannot be contained in, or contain, another range. See exception below.

- Is the full IPv6 prefix if not specified. See exception below.

This exception is for a prefix delegation prefix object with an unspecified range. In this case, the effective range is either:

- The full IPv6 prefix if no other prefixes exist with the same IPv6 prefix, or
- The prefixes that remain when all other ranges for prefix objects with the same IPv6 prefix are removed from the IPv6 prefix.

You create a link only if more than one prefix object with a different IPv6 prefix exists on a link. When the server loads the configuration, if a prefix has no explicit link, the server searches for or creates an implicit link with the name `Link-[vpn.name/]prefix`. All prefix objects with the same IPv6 prefix must either not specify a link or explicitly specify the same link.

The DHCPv6-enabled server supports VPNs (namespaces) for DHCPv6. However there is presently no means to make use of anything other than the default global VPN (there is no VPN option). Both the link and prefix objects have a *vpn-id* attribute, because prefixes do not require links, but all prefixes on a link must use the same VPN ID.

## See Also

[Determining Links and Prefixes](#)  
[Generating Addresses, page 26-4](#)  
[Generating Delegated Prefixes, page 26-4](#)

## Determining Links and Prefixes

When the DHCPv6 server receives a DHCPv6 message, it determines the links and prefixes it uses to service the request. The server:

1. Finds the source address:
  - a. If the client message was relayed, the server sets the source address to the first nonzero link-address field starting with the Relay-Forward message closest to the client (working outwards). If the server finds a source address, it proceeds to step 2.
  - b. Otherwise, if the message source address is a link-local address, the server sets the source address to the first address for the interface on which it received the message for which a prefix exists (or 0 if it finds no prefix for any address). It then proceeds to step 2.
  - c. Otherwise, the server sets the source address to the message source address.
2. Locates the prefix for the source address. If the server cannot find a prefix for the source address, it cannot service the client and drops the request.
3. Locates the link for the prefix. This always exists and is either an explicitly configured link or the implicitly created link based on the prefix address.

Now that the server can determine the client link, it can process the client request. Depending on whether the client request is stateful or prefix-delegated, and on the selection criteria and other factors, the server might use one or more prefixes for the link to service the client request.

This is one area of difference between DHCPv4 and DHCPv6. In DHCPv4, the server selects only one of the scopes from the network to service the client request. In DHCPv6, the server can use all the prefixes for the link. Thus, the server might assign a client an address, or delegate a prefix, from multiple prefixes for the link (subject to selection criteria and other conditions).

## Generating Addresses

IPv6 addresses are 128-bit addresses (as compared to 32-bit addresses for IPv4). In most cases, DHCPv6 servers assign 64 of those bits, the interface-identifier (EUI-64) portion (see RFC 4291). You can generate addresses by using the client 64-bit interface-identifier or a random number generator. The interface-identifier emulates how stateless autoconfiguration assigns addresses to clients. Unfortunately, there are privacy concerns regarding its use, and it is limited to one address per prefix for the client.

By default, Cisco Network Registrar generates an address using an algorithm similar to that described in RFC 4941 to generate a random interface identifier. These random interface identifiers have a zero value for the universal/local bit to distinguish them from EUI-64-based identifiers. The server also skips randomly generated interface identifiers from `::0` to `::ff` so that you can use identifiers for infrastructure devices (such as routers). You can configure whether to assign the interface-identifier (if available) first for each prefix (through the interface-identifier flag of the prefix *allocation-algorithms* attribute). (See the [“Creating and Editing Prefixes” section on page 26-24](#).) If you specify use of the interface-identifier, the server might still use randomly generated addresses if the address is not available to the client, or the client requests multiple addresses on a prefix.

The server generates addresses based on the prefix-configured range (or the prefix address if there is no range). If the range prefix length is shorter than 64, the server supplies only 64 bits and places them in the address interface-identifier field. If the prefix length is longer than 64, the server supplies only the remaining bits of the address. Thus, a /96 range uses 96 bits from the specified range followed by 32 bits of either the client interface-identifier or a randomly generated value. If the resulting address is not available (such as if it is already leased to another client, or to the same client, but on a different binding), the server tries to generate another address. It repeats this process up to at most 500 times.

**Note**

The DHCP server tests only the randomly generated interface identifier for values from `::0` to `::ff`, not the resulting address. Thus, a randomly generated address may end up using an `xxxx:xxxx:xxxx:xxxx::0` through `xxxx:xxxx:xxxx:xxxx::ff` address if the length of the prefix is longer than /64 and the prefix bits that extend beyond the /64 boundary are all zero.

**Tip**

You can also choose from additional address generation algorithms for a prefix and prefix template; see the [“Creating and Editing Prefix Templates” section on page 26-18](#).

## Generating Delegated Prefixes

The DHCPv6 server uses the best first-fit algorithm when generating delegated prefixes. The server uses the first longest available prefix of the length configured or requested.

## DHCPv6 Clients and Leases

The DHCPv6 server supports clients and leases that are similar to those for DHCPv4. The key differences are:

- The server identifies DHCPv6 clients by their DHCP Unique Identifier (DUID), which is the DHCPv4 concept of hardware addresses and client IDs consolidated into one unique client identifier.
- DHCPv6 clients can have multiple leases. This means that if multiple prefixes are on a single link, the server assigns the client an address from each prefix that it is allowed to use, not just from one scope, as in DHCPv4.
- The server first creates a DHCPv6 client when it associates the first lease with it, and deletes the client when it no longer has any leases associated with it. This is identical to DHCPv4 behavior, except that a DHCPv4 client can only have a single lease.
- DHCPv6 leases are dynamically created. The server does not create all leases that it can potentially use at configuration time, because there potentially could be billions of these leases.

Leases can be for:

- **Nontemporary addresses**—Standard IPv6 unicast addresses with likely long (and renewable) lifetimes.
- **Temporary addresses**—Standard IPv6 unicast addresses, but with very limited (and nonrenewable) lifetimes. Temporary addresses solve a privacy issue with IPv6 (see RFC 3041).
- **Delegated prefixes**—Used for prefix delegation (see RFC 3633).

Leases have both a preferred and valid lifetime:

- **Preferred lifetime**—Primarily for the use of the client, the length of time that a valid address is preferred. When the preferred lifetime expires, the address becomes deprecated.
- **Valid lifetime**—Used by both client and server, it is the length of time an address remains in the valid state. The valid lifetime must be greater than or equal to the preferred lifetime. When the valid lifetime expires, the address becomes invalid. A lease is eligible to be deleted once the valid lifetime expires. This is essentially the same as the DHCPv4 lease time.

### See Also

[DHCPv6 Bindings](#)

[Lease Affinity](#)

[Lease Life Cycle, page 26-6](#)

[DHCPv6 Reservations, page 26-7](#)

[Searching for Leases, page 26-9](#)

[Querying Leases for DHCPv6, page 26-9](#)

## DHCPv6 Bindings

Bindings are new to DHCPv6 and allow multiple groups of addresses to be allocated to a client. A client binding consists of one of three types:

- Nontemporary (IA\_NA)
- Temporary (IA\_TA)
- Prefix delegation (IA\_PD)

A binding also consists of a unique Identity Association Identifier (IAID). Leases always exist under a binding. Clients, therefore, have one or more bindings, and bindings have one or more leases. The server creates bindings when it first adds the lease, and removes the binding when it has no more leases. The server creates clients when adding the first binding, and removes them when it has no more bindings.

## Lease Affinity

For DHCPv4, when a lease expires or the server releases it, the server remembers the client for an address as long as it is not assigned to another client. For DHCPv6, because of the large IPv6 address space and depending on the address generation technique, eons could pass before an address needs reassignment to another client. Therefore, Cisco Network Registrar provides an *affinity-period* attribute so that the client can get the same address even if not requesting a renewal before expiration.

The affinity period is desirable in some environments, but not in others where the affinity time would be zero or very small. During the affinity period, the lease is in the AVAILABLE state and still associated with the client that last leased it. If the client requests a lease during this period, the server grants it the same lease (or, if renewals are inhibited, the client explicitly does not get that lease).

## Lease Life Cycle

Leases have a life cycle controlled by states. A lease only exists while it is associated with a client and the server deletes it once it is no longer associated with that client. The life cycle and state transitions are:

1. A lease is born and associated with an address when the server:
  - a. Creates a reservation for a lease, which puts the lease in the AVAILABLE state and marks it as RESERVED. No timer is associated with this state and the server does not delete the lease as long as it is RESERVED.
  - b. Sends an ADVERTISE message to a client, which puts the lease in OFFERED state. The lease transitions to DELETED state after the offer timeout.
  - c. Sends a REPLY message to a client (for a REQUEST, RENEW, or REBIND), which puts the lease in LEASED state. The lease transitions to EXPIRED state after the valid lifetime for the lease elapses.
2. An OFFERED lease transitions to:
  - a. LEASED state when the server receives a REQUEST message, and then transitions to EXPIRED state after the valid lifetime for the lease elapses.
  - b. DELETED state if the offered-time expires.
3. A LEASED lease:
  - a. Is renewed when the server receives a REQUEST, RENEW, or REBIND message. The lease transitions to EXPIRED state after the new valid lifetime for the lease elapses (note that the new valid lifetime could be 0).

- b. Transitions to RELEASED state when the server receives a RELEASE message. The lease transitions to AVAILABLE state after the release-grace-period elapses.
  - c. Transitions to UNAVAILABLE state when the server receives a DECLINE message. The server deletes the lease after the unavailable timeout period elapses.
- 4. An EXPIRED lease transitions to AVAILABLE state after the grace-period. The server deletes the lease after the affinity-period elapses.
- 5. An AVAILABLE lease:
  - a. Transitions to DELETED state and the server deletes it from memory and the lease database after the affinity-period elapses.
  - b. Cannot be deleted if it is RESERVED, and it remains AVAILABLE.
- 6. The server can reoffer a LEASED, EXPIRED, RELEASED, or AVAILABLE lease to a client, but it remains in its current state, although the server extends the timeout to at least the *offer-timeout*.
- 7. A LEASED lease can also transition to REVOKED state if the server needs to revoke the lease. A revoked lease was previously valid but became invalid because of configuration or selection tag changes. The server can revoke a lease when the client attempts to renew, if the lease is reserved for a different client or the prefix is no longer usable. The lease transitions to AVAILABLE again only after its valid lifetime expires or the client sends a SOLICIT for a new lease.

## DHCPv6 Reservations

Reservations apply to nontemporary addresses and delegated prefixes only. They are associated with a prefix in the configuration, and must always be for an address (or prefix) under a configured prefix object.

The reservation can be outside the object range of the prefix, provided it is not in object range of another prefix. However, this has implications when you add a new prefix object. If a reservation that is contained in the new range of the prefix exists, the prefix will not be added. This results in an EX\_CONFLICT status. For details, see the [“Creating Lease Reservations” section on page 22-14](#).



### Note

The operations for DHCPv4 reservations are similar to DHCPv6 reservations, except that the addresses are v6 addresses, not v4 addresses. Also, the main identity for a DHCPv6 client is a client DUID, and not the mac-address. DHCPv6 reservations include addresses and delegated prefixes.

Any change you make in the v6 reservation list modifies the parent prefix to indicate that a server reload is required. On the regional server, if the DHCP edit mode is synchronous and the parent prefix has been assigned to a local cluster, changes are automatically forwarded to the local cluster. A server reload is required, before these changes take effect.



### Caution

If multiple DHCP servers distribute IP addresses on the same subnet, the client reservations must be identical. If not, a client for whom a lease reservation exists can receive offers of different IP addresses from different servers.

A lease reservation pairs an IP address with a lookup key. A lookup key can be a string value or binary blob. You can choose any valid IP address on the network; it does not necessarily have to be in one of the scope ranges. You can use the IP addresses in the scope range for dynamic leases and the ones outside the range for reserved leases. However, even though a reserved IP address may not be in the scope range, the policy associated with the scope still applies to the address.

## Local Advanced Web UI

To view the reservations for DHCPv6 prefixes, do the following:

- 
- Step 1** To view DHCPv6 lease reservations, choose **Prefixes** from the **DHCPv6** menu to open the List/Add DHCPv6 Prefixes page.
  - Step 2** Enter Prefix Name and Prefix Address. Enter value for Range and then choose values for DHCP Type, Template, Owner and Region from the respective drop-down lists. Click **Add Prefix**.
- 

To configure the reservations directly for DHCPv6 prefixes, do the following:

In the advanced mode, if a valid parent prefix is not specified, the CCM server automatically sets the appropriate parent prefix.

- 
- Step 1** From the **DHCP v6** menu, choose **Reservations** to open the List/Add DHCP v6 Reservations page.
  - Step 2** To create a reservation on this page, enter the IP address you want to reserve for lease, and enter a lookup key in the Lookup Key field.
  - Step 3** Click the **String** radio button, if you entered string value or click the **Binary** radio button, if you entered binary value in the Lookup Key field.
  - Step 4** Click **Add Reservation**.
  - Step 5** Choose a filter type from the Filter Type drop-down list. Enter a value in the Filter Value field. Click **Set Filter**. To set Filter Type as None, click **Clear Filter**.
- 

The lease IP address, Lookup Key and Prefix details are displayed in the List/Add DHCPv6 Reservations page.

---

## CLI Commands

The reservation6 command lets you access Cisco Network Registrar's global list of DHCPv6 reservations.

A matching prefix must exist for each reservation in the global list, otherwise the edit is rejected as invalid.

Create a new address by using, **reservation6** [vpn-name/]address **create** lookup-key [blob | string] [attribute=value]

For example:

```
nrcmd> reservation6 red/172.16.0.1 create 172.30.10.1 BlobGreen.htm scope=100
```

Delete an address by using, **reservation6** [vpn-name/]address **delete**

For example:

```
nrcmd> reservation6 white/172.16.0.1 delete
```

Get an address by using, **reservation6** [vpn-name/]address **get** value

For example:

```
nrcmd> reservation6 white/172.16.0.1 get value
```

Set an attribute by using, **reservation6** [vpn-name/]address **set** scope=value



For example:

```
nrcmd>reservation6 white/172.16.0.1 set scope=200
```

Unset an attribute by using, **reservation6** [vpn-name/]address **unset** value

For example:

```
nrcmd>reservation6 white/172.16.0.1 unset value
```

Show an address by using, **reservation6** [vpn-name/]address **show**

For example:

```
nrcmd>reservation6 white/172.16.0.1 show
```

## Searching for Leases

For details on searching for leases in the configured DHCPv6 network, see the [“Searching Server-Wide for Leases”](#) section on page 22-9.

## Querying Leases for DHCPv6

For details on the DHCPLEASEQUERY implementation for DHCPv6, see the [“Leasequery for DHCPv6”](#) section on page 22-34.

## DHCPv6 Policy Hierarchy

DHCPv6 uses the existing policy objects, with additional DHCPv6 specific attributes (that are mostly analogous to those in DHCPv4). For DHCPv6, the hierarchy is:

1. Client embedded policy
2. Client named policy
3. Client-class embedded policy
4. Client-class named policy
5. Prefix embedded policy
6. Prefix named policy
7. Link embedded policy
8. Link named policy
9. system\_default\_policy

For attributes, the default value for the most local policy applies. This hierarchy is the same as for DHCPv4, except for the additional link policies and the fact that the prefix policies replace the scope policies. (For a comparison with the DHCPv4 policy hierarchy, see the [“Policy Hierarchy”](#) section on page 21-3.)

The hierarchy applies to most policy attributes, which the server processes in the context of a single prefix. However, the server processes a few attributes (specifically *allow-rapid-commit*, *reconfigure*, *v6-reply-option*, *v6-options*, and *v6-vendor-options*) in the context of multiple prefixes. In these cases, the processing at the prefix levels (steps 5 and 6) is a bit different:

- For the *reconfigure* attribute that controls whether the server requires, allows, or disallows client reconfiguration, the server checks the embedded and named policies of all prefixes on the link that the client is allowed to use (based on selection tags). If any of the prefix policies have the *reconfigure* attribute set to **disallow** or **require**, the server uses that setting. Otherwise, if at least one policy has it set to **allow**, Reconfigure is allowed. Otherwise, the server checks the remaining policies in the hierarchy. (See the “[Reconfigure Support](#)” section on page 26-32 for details.)
- If the client requests Rapid Commit (see the “[Editing DHCPv6 Server Attributes](#)” section on page 26-29), the server checks the embedded and named policies of all prefixes on the link that the client is allowed to use (based on selection tags). If one of these policies has *allow-rapid-commit* disabled, the server processes the client request as if Rapid Commit were not part of the request. If at least one policy has *allow-rapid-commit* enabled, the client can use Rapid Commit. If no prefix policy has the attribute configured, processing continues at step 7.
- For the options-related attributes (see the “[Setting DHCPv6 Options](#)” section on page 26-31), the server also does special handling at steps 5 and 6. The server checks the embedded and then named policy of each prefix on the link. It then uses the first one with the configured *v6-reply-option* attribute, or the first one with the configured value for the *v6-options* or *v6-vendor-options*.
- The server checks the prefixes in case-insensitive alphabetical order.

**Tip**

In configurations with multiple prefixes on a link, avoid setting the Rapid Commit and option properties for the prefix policy, but rather set them on the link policy or other policy instead.

## DHCPv6 Options

DHCPv6 options do not use DHCPv4 options; they are unique and separate. There are currently about 46 DHCPv6 options. Most of these options are the DHCPv6 protocol infrastructure options and are not user-definable. They use a 16-bit option code and 16-bit length (DHCPv4 uses only 8 bits for both of these). Configuring options and the behavior of configured options in policies are similar to those for DHCPv4. See the “[Setting DHCPv6 Options](#)” section on page 26-31 for details about client processing as it relates to the policy hierarchy.

# DHCPv6 Configuration

The following sections describe how to configure DHCPv6 in Cisco Network Registrar:

- [Viewing IPv6 Address Space](#)
- [Configuring Links](#)
- [Configuring Prefixes, page 26-18](#)
- [Viewing DHCPv6 Networks, page 26-29](#)
- [Editing DHCPv6 Server Attributes, page 26-29](#)
- [Configuring DHCPv6 Policies, page 26-29](#)
- [Configuring DHCPv6 Client-Classes, page 26-30](#)
- [Configuring DHCPv6 Clients, page 26-31](#)
- [Setting DHCPv6 Options, page 26-31](#)
- [Reconfigure Support, page 26-32](#)

## Viewing IPv6 Address Space

When you choose **Address Space** from the **Address Space v6** drop-down list, then in the local advanced or regional web UI, you open the View Unified v6 Address Space page. This page is like the View Unified Address Space page for IPv4 (see the [“Viewing Address Space” section on page 9-3](#)). On the View Unified v6 Address Space page you can:

- Set a VPN for the address space.
- Add a prefix by adding its name and address and choosing a DHCP type and possible template. Click **Add Prefix** to open the Add Prefix page (see the [“Creating and Editing Prefixes” section on page 26-24](#)).
- Edit a prefix by clicking its name. This opens the Edit Prefix page (see the [“Creating and Editing Prefixes” section on page 26-24](#)).
- View the current usage of the prefix space (see the [“Viewing Address Utilization for Prefixes” section on page 26-27](#)).

## Configuring Links

You can configure DHCPv6 links directly, or you can create link templates for them first. See the following subsections:

- [Creating and Editing Link Templates](#)
- [Creating and Editing Links, page 26-17](#)

### Creating and Editing Link Templates

You can create links from predefined templates. The attributes you can set for a link template are as follows (for the expression syntax, see the [“Using Expressions in Link Templates”](#) section on [page 26-14](#)):

- *name*—User-assigned name for the link template.
- *description*—Description of the link template itself.
- *policy*—Shared policy used when replying to clients, as applied to the link.
- *link-name-expr*—Expression to define the name of the link once the template is applied.
- *link-description-expr*—Expression to define the description on the link once applied.
- *prefix-expr*—Expression to create the list of associated prefixes once the template is applied. For example, you can specify creating prefixes based on defining *prefix-expr* as **@link-prefix-expr.txt** to point to the file that contains this expression (and assuming that the *cm-prefix*, *cpe-address-prefix*, and *cpe-pd-prefix* templates exist):

```
(list
(create-prefix "cm-prefix" (create-prefix-range 32 0x1))
(create-prefix "cpe-address-prefix" (create-prefix-range 32 0x2))
(create-prefix "cpe-pd-prefix" (create-prefix-range 16 0x1))
)
```
- *options-expr*—Expression to define the list of embedded policy options to create with the link.

#### Local Advanced and Regional Web UI

- 
- |               |   |
|---------------|---|
| <b>Step 1</b> | From the <b>DHCPv6</b> menu, choose <b>Link Templates</b> . The List/Add DHCPv6 Templates page appears. The page displays the existing templates.   |
| <b>Step 2</b> | Click <b>Add Link Template</b> to open the Add DHCPv6 Link Template page.   |
| <b>Step 3</b> | Enter a link template name, optional description, and optionally choose a preconfigured policy from the drop-down list.   |
| <b>Step 4</b> | Add expressions for the <i>link-name-expr</i> , <i>link-description-expr</i> , <i>prefix-expr</i> , or <i>options-expr</i> field attributes (see the <a href="#">“Using Expressions in Link Templates”</a> section on <a href="#">page 26-14</a> ). |
| <b>Step 5</b> | Click <b>Add Link Template</b> .  |

**Step 6** In the regional web UI, you can pull replica link templates or push templates to local clusters:

- Click **Pull Replica Link Template** to open the Select DHCPv6 Link Template Data to Pull page. Choose a pull mode for the cluster (ensure, replace, or exact), then click **Pull All Link Templates**. On the Report Pull DHCPv6 Link Template page, click **OK**.
- Click **Push Link Template** for a specific template (or **Push All Link Templates**) to open the Push DHCPv6 Link Template Data to Local Cluster page. Choose a data synchronization mode (ensure, replace, or exact), move the desired cluster or clusters to the Selected table, then click **Push Data to Clusters**.

## CLI Commands

To create the link template, use **link-template name create**. For example:

```
nrcmd> link-template example-link-template create [attribute=value]
```

You can set and enable the aforementioned expression setting attributes in the usual way, and you can show and list link templates. For example, to set a prefix expression for the link template, use the following file definition and pointer to the file (and assuming that the cm-prefix, cpe-address-prefix, and cpe-pd-prefix templates exist):

```
> type link-prefix-expr.txt
(list (create-prefix "cm-prefix" (create-prefix-range 32 0x1))
 (create-prefix "cpe-address-prefix" (create-prefix-range 32 0x2))
 (create-prefix "cpe-pd-prefix" (create-prefix-range 16 0x1)) )
```

```
nrcmd> link-template example-link-template set prefix-expr=@link-prefix-expr.txt
```

In addition:

- To clone a link template, use **link-template name create clone=name**.
- To apply a template to one or more links, use **link-template name apply-to {all | link[,link,...]}**. You can create prefixes by using **link-template name apply-to link [prefix]**, but only with one link specified.
- The link-template includes an embedded-policy object. The link-template-policy CLI command and the Web UI supports the embedded policy on the link-template page.

## Using Expressions in Link Templates

You can specify expressions in a link template to dynamically create prefix names, IP address ranges, and embedded options when creating a link. Expressions can include context variables and operations.



### Note

Expressions are not the same as DHCP extensions. Expressions are commonly used to create client identities or look up clients. Extensions (see [Chapter 29, “Using Extension Points”](#)) are used to modify request or response packets.

When a template is applied to a link, if the link-template has an embedded policy, it is copied to the link. This embedded policy may or may not have options. As the entire link-template's embedded policy is used (if it exists), it will wipe out any existing options in the link. If the link-template has no embedded policy, the link's embedded policy is retained. Next the link-template's option expression, if any, is evaluated and the options are added to the embedded policy options in the link (if no embedded policy exists, one is created).

[Table 26-1](#) lists the link template predefined variables and [Table 26-2](#) lists the link template operators. Note that these variables and operators are not case-sensitive. [Table 26-4](#) lists the prefix template operators. The link template operators table and prefix template operations table both have same operators, except that only a link template can use **Create Prefix Operator** and prefix template can not use the operator.

**Table 26-1** Link Template Expression Predefined Variables

Predefined Variable	Description
<b>mask-length</b>	Number of prefix mask bits (with a <i>template-root-prefix</i> defined).
<b>prefix</b>	Network number and length (with a <i>template-root-prefix</i> defined).
<b>prefix-addr</b>	Address portion of the prefix (with a <i>template-root-prefix</i> defined).
<b>prefix-length</b>	Number of prefix address bits (with a <i>template-root-prefix</i> defined).
<b>template.attribute</b>	Attribute of the link template.
<b>vpn</b>	VPN of the link.



**Table 26-2** Link Template Expression Operators

Expression Operator	Description
<b>Arithmetic Operators</b> (unsigned integer arguments only)	
<b>(+ arg1 arg2)</b>	Adds the two argument values, such as (+ 2 3).
<b>(– arg1 arg2)</b>	Subtracts the second argument value from the first one.
<b>(* arg1 arg2)</b>	Multiplies the values of two arguments.
<b>(/ arg1 arg2)</b>	Divides the value of the first argument by that of the second one (which cannot be zero).
<b>(% arg1 arg2)</b>	Modulo arithmetic operator to determine the remainder of the result of the first argument divided by the second one.
<b>Concatenation Operator</b>	
<b>(concat arg1 ... argn)</b>	Concatenates the arguments into a string.

Table 26-2 Link Template Expression Operators (continued)

Expression Operator	Description
<b>List Operator</b>	
( <b>list</b> <i>oper1</i> ... <i>opern</i> )	<p>Creates an options list or list of prefixes. Required if you need more than one option for a link or prefix, or more than one prefix for a link. All arguments must be <b>create-v6-option</b> operation. Nesting is not supported. For example:</p> <pre>(list (create-prefix "cm-prefix" (create-prefix-range 32 0x1))       (create-prefix "cpe-address-prefix" (create-prefix-range 32 0x2))       (create-prefix "cpe-pd-prefix" (create-prefix-range 16 0x1)) )</pre>
<b>Create Prefix Operator</b>	
( <b>create-prefix</b> <i>template</i> <i>prefix</i> )	<p>Creates a prefix based on a predefined prefix template name and the prefix, including the link VPN (assuming that a <i>template-root-prefix</i> is defined).</p> <p>The <i>prefix</i> argument can be the prefix name, but also the <b>create-prefix-addr</b> or <b>create-prefix-range</b> operator value. You can use the <b>list</b> function to combine multiple operations. For example:</p> <pre>(create-prefix "cm-prefix" (create-prefix-range 32 0x1))</pre>
<b>Create IP Operator</b>	
( <b>create-prefix-addr</b> <i>prefix</i> <i>interface-id</i> )	<p>Creates an IPv6 address string (assuming that a <i>template-root-prefix</i> is defined) based on the prefix name and interface ID (an IPv6 address that you can specify as a string), which is the lower 64-bit address in the prefix (which need not be contained in the parent prefix). Used in the <i>prefix-expr</i> and <i>options-expr</i> attributes.</p>
<b>Create Range Operator</b>	
( <b>create-prefix-range</b> <i>size</i> <i>n</i> )	<p>Creates an address range (child) for the prefix, used in the <i>prefix-expr</i> attribute. The prefix value that the function is based on is either the <i>template-root-prefix</i> if applying a link template to a link, or the prefix address, if applying a prefix template to a prefix.</p> <p>Range value—An increase in the prefix length.</p> <p>Size—The number of bits by which you can increase the prefix length. Must be a value from 1 through 32. Must be less than the parent prefix length.</p> <p>n—The nth occurrence of the child prefix. Value can be 0, but is limited to less than two to the power of the size. Must be less than or equal to the size.</p> <p>The size and n must be greater than zero.</p> <p>The <i>n</i> must be less than or equal to the <i>size</i>, and the <i>size</i> must be less than the parent prefix length. For example:</p> <pre>(create-prefix-range 32 0x1)</pre>

Table 26-2 Link Template Expression Operators (continued)

Expression Operator	Description
<b>Create Option Operator</b>	
<b>(create-option <i>opt val</i>)</b>	<p>Creates a DHCPv6 option, used in the <i>options-expr</i> attribute. The <i>opt</i> can be the literal string or integer identifying the option. The <i>val</i> is the string representation of the option value, as defined by the option TLV value.</p> <p>You can use custom defined and unknown options. For undefined options, the option number must be specified and the data is used as is (as blob data). If the data is a string, the string is used as is and if the data is a number or address, it is used as is.</p> <p>For example:</p> <pre>(list (create-option "dns-servers" (create-prefix-addr prefix "::2")) (create-option "domain-list" "sales.example.com,example.com"))</pre> 
<b>Note</b>	<b>(create-v6-option <i>opt val</i>)</b> is a synonym for <b>(create-option)</b> and can be used instead; but we recommend that you use <b>(create-option)</b> .
<b>Create Vendor Option Operation</b>	
<b>(create-vendor-option <i>set-name opt val</i>)</b>	<p>Creates a DHCPv6 vendor option, used in the <i>options-expr</i> attribute. The <i>set-name</i> specifies the option definition set for the vendor option. The <i>opt</i> can be the literal string or integer identifying the vendor option in the set. The <i>val</i> is representation of the option value.</p> <p>For example:</p> <pre>(list (create-option "dns-servers" (create-prefix-addr prefix "::2")) (create-vendor-option "dhcp6-cablelabs-config" 17 "(enterprise-id 4491((tftp-servers 32 3800:0:0:180::6) (config-file-name 33 modem_ipv6.bin)(syslog-servers 34 3800:0:0:180::8) (rfc868-servers 37 3800:0:0:180::6)(time-offset 38 -5h) (cablelabs-client-configuration 2170 (primary-dhcp-server 1 10.38.1.5) (secondary-dhcp-server 2 10.38.1.6))))))</pre> 
<b>Note</b>	<b>(create-v6-vendor-option <i>opt val</i>)</b> is a synonym for <b>(create-vendor-option)</b> and can be used instead; but we recommend that you use <b>(create-vendor-option)</b> .



## Creating and Editing Links

You can create links directly. The attributes you can set for the link are:

- **name**—User-assigned name for the link.
- **vpn-id**—VPN that contains the link.
- **description**—Descriptive text for the link.
- **policy**—Shared policy used when replying to clients.
- **free-address-config**—Identifies which trap captures unexpected free address events on this prefix. If not configured, the server looks at its *v6-default-free-address-config* attribute.

### Local Advanced and Regional Web UI

- 
- Step 1** From the **DHCP v6** menu, choose **Links**. The List/Add DHCP v6 Links page displays the existing links.
- Step 2** To add a link, enter the desired name for the link and click **Add Link**.  
Click **Add Link** without entering any values in the List/Add DHCP v6 Links page to view the Add DHCPv6 Link page. In the Add DHCPv6 Link page, you can enter the desired name for the link and also set other attributes like optional description, policy, owner, region, and free-address-config.
- Step 3** Choose the predefined prefixes for the link by moving them to the Selected field.
- Step 4** To add new prefixes for the link, enter each prefix name and its address at the bottom of the page, indicate a range, choose the DHCP type and template (if needed), then click **Add Prefix** for each one.
- Step 5** Click **Add Link**.
- Step 6** In the regional web UI, you can push links and their associated prefixes to local clusters. Click **Push Link** for a specific link (or **Push All Links**) to open the Push DHCPv6 Link Data to Local Cluster page. Choose a data synchronization mode (ensure, replace, or exact), move the desired cluster or clusters to the Selected table, then click **Push Data to Clusters**.
- 

### CLI Commands

Use **link name create**. (The **link** command is a synonym for the **dhcp-link** command from previous releases.) For example:

```
nrcmd> link example-link create [attribute=value]
```

To apply a link template during link creation, use **link name create template=name [template-root-prefix=address]**, with the *template-root-prefix* specified if the template could create more than one prefix. To apply a template to an existing link definition, use **link name applyTemplate template-name [template-root-prefix]**.

You can set and enable the aforementioned attributes in the usual way, and you can show and list links. To list prefixes or prefix names associated with a link, use **link name listPrefixes** or **link name listPrefixNames**.

## Configuring Prefixes

You can configure DHCPv6 prefixes directly, or you can create prefix templates for them first. See the following subsections:

- [Creating and Editing Prefix Templates](#)
- [Creating and Editing Prefixes, page 26-24](#)
- [Viewing Address Utilization for Prefixes, page 26-27](#)

### Creating and Editing Prefix Templates

You can create prefixes from predefined templates. The attributes you can set for a prefix template are the following (for the expression syntax, see the [“Using Expressions in Prefix Templates”](#) section on page 26-21):

- **name**—User-assigned name for the prefix template.
- **description**—Descriptive text for the prefix template.
- **dhcp-type**—Defines how DHCP manages address assignment for a prefix:
  - **dhcp** (preset value)—Uses the prefix for stateful address assignment.
  - **stateless**—Uses the prefix for stateless option configuration.
  - **prefix-delegation**—Uses the prefix for prefix delegation.
  - **infrastructure**—Uses the prefix to map a client address to a link, when the prefix does not have an address pool.
- **policy**—Shared policy to use when replying to clients.
- **prefix-name-expr**—Expression that evaluates to a string value to use for the name of the prefix created. For example, you can have the prefix name prepended by **CM-** if you define *prefix-name-expr* as (**concat "CM-" prefix**). In the CLI, you would include the expression in a file and point to that file:

```
> type prefix-name.txt
(concat "CM-" prefix)
```

```
nrcmd> prefix-template ex-template create prefix-name-expr=@prefix-name.txt
```

- **prefix-description-expr**—Expression that evaluates to a string value to apply to the description on the prefix created when using the template.
- **range-expr**—Expression that evaluates to an IPv6 prefix value to create an address range. In the CLI, you must use a file reference. For example:

```
> type subprefix-expr.txt
(create-prefix-range 1 0x1)
```

```
nrcmd> prefix-template ex-template set range-expr=@subprefix-expr.txt
```

- **options-expr**—Expression that evaluates to embedded policy options to create. (Use the **list** function to create multiple options.)
- **allocation-algorithms**—One or more algorithms the server uses to select a new address or prefix to lease to a client. The available algorithms are:
  - **client-request** (preset to off)—Controls whether the server uses a client-requested lease.

- **reservation** (preset to on)—Controls whether the server uses an available reservation for the client.
- **extension** (preset to on)—Controls whether the server calls extensions attached at the **generate-lease** extension point to generate an address or prefix for the client. For details on extensions, see [Chapter 29, “Using Extension Points.”](#)
- **interface-identifier** (preset to off)—Controls whether the server uses the interface-identifier from the client (link-local) address to generate an address; ignored for temporary addresses and prefix delegation.
- **random** (preset to on)—Controls whether the server generates an address using an RFC 3041 algorithm; ignored for prefix delegation.
- **best-fit** (preset to on)—Controls whether the server delegates the first, best-fit available prefix; ignored for addresses.

When the server needs an address to assign to a client, it processes the flags in the following order until it finds a usable address: client-request, reservation, extension, interface-identifier, and random. When the server needs to delegate a prefix to a client, it processes the flags in the following order until it finds a usable prefix: client-request, reservation, extension, and best-fit.

- **max-leases**—Maximum number of nonreserved leases allowed on the prefix. When a new lease needs to be created, the server does so only if the limit is not exceeded. When the limit is exceeded, the server cannot create or offer new leases to clients. If you also enable SNMP traps, the *max-leases* value also calculates the percentage of used and available addresses.



**Note** Be sure to set the *max-leases* value to the expected maximum so that the SNMP address traps can return meaningful results.

- **ignore-declines**—Controls whether the server responds to a DHCPv6 DECLINE message that refers to an IPv6 address or a delegated prefix from this prefix. If enabled, the server ignores all declines for leases in this prefix. If disabled (the preset value) or unset, the server sets to UNAVAILABLE every address or delegated prefix requested in a DECLINE message if it is leased to the client.
- **deactivated**—Controls whether a prefix extends leases to clients. A deactivated prefix does not extend leases to any clients and treats all addresses in its ranges as if they were individually deactivated. The preset value is false (activated).
- **expiration-time**—Time and date at which a prefix expires. After this date and time, the server neither grants new leases nor renews existing leases from this prefix. Once the *expiration-time* passes, the prefix is no longer used (although old leases and leases with grace or affinity periods continue to exist until those periods elapse). Enter a value in the format "[weekday] month day hh:mm[:ss] year"; for example, "**Dec 31 23:59 2006**".
- **reverse-zone-prefix-length**—Prefix length of the reverse zone for ip6.arpa updates. (See the [“Determining Reverse Zones for DNS Updates”](#) section on page 28-4 for details.)
- **selection-tags**—List of selection tags associated with the prefix.

## Local Advanced and Regional Web UI

- 
- Step 1** From the **DHCPv6** menu, choose **Prefix Templates**. The List/Add DHCP Prefix Templates page shows the existing templates.
- Step 2** Click **Add Prefix Template** to open the Add DHCPv6 Prefix Template page.
- Step 3** Set the attributes and add expressions for the templates that require expressions (see the [“Using Expressions in Prefix Templates”](#) section on page 26-21).
- Step 4** Click **Add Prefix Template**.
- Step 5** To edit a prefix template, click its name on the List/Add DHCPv6 Prefix Template page. On the Edit DHCPv6 Prefix Template page, edit the template attributes, such as adding a selection tag, then click **Modify Prefix Template**.
- Step 6** In the regional web UI, you can pull replica prefix templates or push templates to local clusters:
- Click **Pull Replica Prefix Template** to open the Select DHCPv6 Prefix Template Data to Pull page. Choose a pull mode for the cluster (ensure, replace, or exact), then click **Pull All Prefix Templates**. On the Report Pull DHCPv6 Prefix Template page, click **OK**.
  - Click **Push Prefix Template** for a specific template (or **Push All Prefix Templates**) to open the Push DHCPv6 Prefix Template Data to Local Cluster page. Choose a data synchronization mode (ensure, replace, or exact), move the desired cluster or clusters to the Selected table, then click **Push Data to Clusters**.
- 

## CLI Commands

To create the prefix template, use **prefix-template name create**. For example:

```
nrcmd> prefix-template example-prefix-template create [attribute=value]
```

You can set and enable the aforementioned attributes in the usual way, and you can show and list prefix templates. In addition:

- To clone a prefix template, use **prefix-template name create clone=name**.
- To apply a template to one or more prefixes, use **prefix-template name apply-to {all | prefix[,prefix,...]}**.
- The prefix-template includes an embedded-policy object. The prefix-template-policy CLI command and the Web UI supports the embedded policy on the prefix-template page.

## Using Expressions in Prefix Templates

You can specify expressions in a prefix template to dynamically create prefix names, IP address ranges, and embedded options when creating a prefix. Expressions can include context variables and operations.

**Note**

Expressions are not the same as DHCP extensions. Expressions are commonly used to create client identities or look up clients. Extensions (see [Chapter 29, “Using Extension Points”](#)) are used to modify request or response packets.

When a template is applied to a prefix, if the prefix-template has an embedded policy, it is copied to the prefix. This embedded policy may or may not have options. As the entire prefix-template's embedded policy is used (if it exists), it will wipe out any existing options in the prefix. If the prefix-template has no embedded policy, the prefix's embedded policy is retained. Next the prefix-template's option expression, if any, is evaluated and the options are added to the embedded policy options in the prefix (if no embedded policy exists, one is created).

[Table 26-3](#) lists the prefix template predefined variables and [Table 26-4 on page 26-22](#) lists the operators. Note that these variables and operators are not case-sensitive.



**Table 26-3**      *Prefix Template Expression Predefined Variables*

Predefined Variable	Description
<b>prefix</b>	Network number and length, based on the template root prefix if applying a link template to a link, or the prefix address if applying a prefix template to a prefix.
<b>vpn</b>	VPN of the prefix.
<b>prefix-addr</b>	Address portion of the prefix.
<b>prefix-length</b>	Number of prefix address bits.
<b>mask-length</b>	Number of prefix mask bits.
<b>template.attribute</b>	Attribute of the prefix template.

**Table 26-4 Prefix Template Expression Operators**

Expression Operator	Description
<b>Arithmetic Operators</b> (unsigned integer arguments only)	
(+ <i>arg1 arg2</i> )	Adds the two argument values, such as (+ 2 3).
(− <i>arg1 arg2</i> )	Subtracts the second argument value from the first one, such as with <i>ping-timeout</i> defined as 100, (− template.ping-timeout 10) yields 90.
(* <i>arg1 arg2</i> )	Multiplies the values of two arguments.
(/ <i>arg1 arg2</i> )	Divides the value of the first argument by that of the second one (which cannot be zero).
(% <i>arg1 arg2</i> )	Modulo arithmetic operator to determine the remainder of the result of the first argument divided by the second one.
<b>Concatenation Operator</b>	
(concat <i>arg1 ... argn</i> )	Concatenates the arguments into a string.
<b>List Operator</b>	
(list <i>oper1 ... opern</i> )	Creates an options list or list of prefixes. Required if needing more than one option for a prefix. All arguments must be <b>create-v6-option</b> or <b>create-prefix-range</b> operations. Nesting is not supported.
<b>Create IP Operator</b>	
(create-prefix-addr <i>prefix-name interface-id</i> )	Creates an IPv6 address string based on the prefix name and interface ID (an IPv6 address that you can specify as a string), which is the lower 64-bit address in the prefix (which need not be contained in the parent prefix). Used in the <i>range-expr</i> and <i>options-expr</i> attributes.
<b>Create Range Operator</b>	
(create-prefix-range <i>size n</i> )	<p>Creates an address range (child) for the prefix, used in the <i>range-expr</i> attribute. The prefix value that the function is based on is either the <i>template-root-prefix</i> if applying a link template to a link, or the prefix address if applying a prefix template to a prefix.</p> <p>Range value—An increase in the prefix length.</p> <p>Size—The number of bits by which you can increase the prefix length. Must be a value from 1 through 32. Must be less than the parent prefix length.</p> <p>n—The nth occurrence of the child prefix. Value can be 0, but is limited to less than two to the power of the size. Must be less than or equal to the size.</p> <p>The size and n must be greater than zero. The <i>n</i> must be less than or equal to the <i>size</i>, and the <i>size</i> must be less than the parent prefix length. For example:</p> <p>(create-prefix-range 32 0x1)</p>

Table 26-4 Prefix Template Expression Operators (continued)

Expression Operator	Description
<b>Create Option Operation</b>	
<b>(create-option <i>opt val</i>)</b>	<p>Creates a DHCPv6 option, used in the <i>options-expr</i> attribute. The <i>opt</i> can be the literal string or integer identifying the option. The <i>val</i> is the string representation of the option value, as defined by the option TLV value.</p> <p>You can use custom defined and unknown options. For undefined options, the option number must be specified and the data is used as is (as blob data). If the data is a string, the string is used as is and if the data is a number or address, it is used as is.</p> <p>For example:</p> <pre>(list (create-option "dns-servers" (create-prefix-addr prefix "::2")) (create-option "domain-list" "sales.example.com,example.com"))</pre>  <p><b>Note</b> (create-v6-option <i>opt val</i>) is a synonym for (create-option) and can be used instead.</p>
<b>Create Vendor Option Operator</b>	
<b>(create-vendor-option <i>set-name opt val</i>)</b>	<p>Creates a DHCPv6 vendor option, used in the <i>options-expr</i> attribute. The <i>set-name</i> specifies the option definition set for the vendor option. The <i>opt</i> can be the literal string or integer identifying the vendor option in the set. The <i>val</i> is representation of the option value.</p> <p>For example:</p> <pre>(list (create-option "dns-servers" (create-prefix-addr prefix "::2")) (create-vendor-option "dhcp6-cablelabs-config" 17 "(enterprise-id 4491((tftp-servers 32 3800:0:0:180::6) (config-file-name 33 modem_ipv6.bin)(syslog-servers 34 3800:0:0:180::8) (rfc868-servers 37 3800:0:0:180::6)(time-offset 38 -5h) (cablelabs-client-configuration 2170 (primary-dhcp-server 1 10.38.1.5) (secondary-dhcp-server 2 10.38.1.6))))))</pre>  <p><b>Note</b> (create-v6-vendor-option <i>opt val</i>) is a synonym for (create-vendor-option) and can be used instead.</p>

**Note**

We recommend that you use create-option and create-vendor-option for v4 and v6.

## Creating and Editing Prefixes

You can create prefixes directly (and optionally apply an existing template to it; see the [“Creating and Editing Prefix Templates”](#) section on page 26-18). These are the prefix attributes that you can set:

- **name**—Assigns a name to this prefix.
- **vpn-id**—VPN that contains the prefix.
- **description**—Describes the prefix.
- **dhcp-type**—Defines how DHCP manages address assignment for a prefix:
  - **dhcp** (preset value)—Uses the prefix for stateful address assignment.
  - **stateless**—Uses the prefix for stateless option configuration.
  - **prefix-delegation**—Uses the prefix for prefix delegation.
  - **infrastructure**—Uses the prefix to map a client address to a link, when the prefix does not have an address pool.
  - **parent**—Do not have DHCP use the prefix. But, use it as a container object to group child prefixes. Parent prefixes appear only in the IPv6 address space listing in the web UI, not in the prefixes listing.
- **address**—Prefix (subnet) to which an interface belongs to, using the high-order bits of an IPv6 address.
- **reverse-zone-prefix-length**—Prefix length of the reverse zone for ip6.arpa updates. (See the [“Determining Reverse Zones for DNS Updates”](#) section on page 28-4 for details.)
- **range**—Subrange the server can use to configure prefixes for address assignment. The prefix used depends on the value set for the *dhcp-type* attribute. If unset, the prefix address applies. This value can specify a longer prefix than the prefix address to limit the range of addresses or prefixes available for assignment. (See the [“Links and Prefixes”](#) section on page 26-2 for details.)
- **link**—Link associated with the prefix (subnet), used to group prefixes that are on a single link.
- **policy**—Shared policy to use when replying to clients.
- **selection-tags**—List of selection tags associated with the prefix.
- **allocation-algorithms**—One or more algorithms the server uses to select a new address or prefix to lease to a client. The available algorithms are:
  - **client-request** (preset to off)—Controls whether the server uses a client requested lease.
  - **reservation** (preset to on)—Controls whether the server uses an available reservation for the client.
  - **extension** (preset to on)—Controls whether the server calls extensions attached at the **generate-lease** extension point to generate an address or prefix for the client. For details on extensions, see [Chapter 29, “Using Extension Points.”](#)
  - **interface-identifier** (preset to off)—Controls whether the server uses the interface-identifier from the client (link-local) address to generate an address; ignored for temporary addresses and prefix delegation.
  - **random** (preset to on)—Controls whether the server generates an address using an RFC 3041 algorithm; ignored for prefix delegation.
  - **best-fit** (preset to on)—Controls whether the server delegates the first, best-fit available prefix; ignored for addresses.



When the server needs an address to assign to a client, it processes the flags in the following order until it finds a usable address: client-request, reservation, extension, interface-identifier, and random. When the server needs to delegate a prefix to a client, it processes the flags in the following order until it finds a usable prefix: client-request, reservation, extension, and best-fit.

- **max-leases**—Maximum number of nonreserved leases allowed on the prefix. When a new lease needs to be created, the server does so only if the limit is not exceeded. When the limit is exceeded, the server cannot create or offer new leases to clients. If you also enable SNMP traps, the *max-leases* value also calculates the percentage of used and available addresses.





**Tip**

Set the *max-leases* value to the expected maximum so that the SNMP address traps can return meaningful results.

- **ignore-declines**—Controls whether the server responds to a DHCPv6 DECLINE message that refers to an IPv6 address or a delegated prefix from this prefix. If enabled, the server ignores all declines for leases in this prefix. If disabled (the preset value) or unset, the server sets to UNAVAILABLE every address or delegated prefix requested in a DECLINE message if it is leased to the client.
- **expiration-time**—Time and date at which a prefix expires. After this date and time, the server neither grants new leases nor renews existing leases from this prefix. Once the *expiration-time* passes, the prefix is no longer used (although old leases and leases with grace or affinity periods continue to exist until those periods elapse). Enter a value in the format "[weekday] month day hh:mm[:ss] year"; for example, "**Dec 31 23:59 2006**".
- **deactivated**—Controls whether a prefix extends leases to clients. A deactivated prefix does not extend leases to any clients and treats all addresses in its ranges as if they were individually deactivated. The preset value is false (activated).
- **free-address-config**—Identifies which trap captures unexpected free address events on this prefix. If not configured, the server looks for the *free-address-config* attribute value for the parent link. If that attribute is not configured, the server looks at its *v6-default-free-address-config* attribute.
- **embedded-policy**—Policy embedded in the prefix.

## Local Advanced and Regional Web UI

- Step 1** From the **DHCPv6** menu, choose **Prefixes**. The List/Add DHCPv6 Prefixes page shows the existing prefixes.
- Step 2** Create the prefix:
  - a. If creating it in other than the current VPN, choose a VPN from the drop-down list.
  - b. Enter a prefix name and address, and choose a prefix length from the drop-down list.
  - c. If you want a range of addresses for the prefix, enter the subnet address and choose a prefix length.
  - d. Choose a DHCP type (see the attribute descriptions at the top of this section). The default is DHCP.
  - e. If you want to apply a preconfigured prefix template, choose it from the drop-down list. (Note that the attribute values of an applied template overwrite the ones set for the prefix.)
  - f. Click **Add Prefix**, which should add the prefix to the list.
  - g. Reload the DHCP server. When you return to the List/Add DHCPv6 Prefixes page, a message indicates how many prefixes are synchronized.

- Step 3** To create a reverse zone from the prefix, click the Create icon () in the Reverse Zone column to open the Create Reverse Zone(s) for Prefix page. On this page, you can select a zone template, click **Report**, then **Run**. Click **Return** to return to the List/Add DHCPv6 Prefixes page. The icon in the Reverse Zone column changes to the View icon (), which you can click to open the List/Add Reverse Zones page.
- Step 4** Once you create a prefix, you can list and manage the leases for the prefix by clicking the View icon () in the Leases column of the List/Add DHCPv6 Prefixes page. This opens the List DHCP Leases for Prefix page. From here, you can list the leases for the client lookup key and manage each lease separately by clicking its name. Click **Return** to return to the List/Add DHCPv6 Prefixes page.
- Step 5** You can list and manage the reservations for the prefix by clicking the View icon () in the Reservations column of the List/Add DHCPv6 Prefixes page. This opens the List/Add DHCP Reservations for Prefix page. Add each reservation IP address and lookup key and whether the lookup key is a string or binary, then click **Add Reservation**. Click **Modify Prefix** to return to the List/Add DHCPv6 Prefixes page.
- Step 6** To edit a prefix, click its name on the List/Add DHCPv6 Prefix page. On the Edit DHCPv6 Prefix page, edit the prefix attributes, or create a new or edit an existing embedded policy. To manage an embedded policy:
- Click **Create New Embedded Policy** or **Edit Existing Embedded Policy** to open the Edit DHCP Embedded Policy for Prefix page.
  - Modify the embedded policy properties (see the “[DHCPv6 Policy Hierarchy](#)” section on page 26-9).
  - Click **Modify Embedded Policy**. The next time the Edit DHCPv6 Prefix page appears, you can edit the embedded policy for the prefix.
  - Click **Modify Prefix**.
- Step 7** In the regional web UI, you can push prefixes to local clusters and reclaim prefixes on the List/Add DHCPv6 Prefixes page:
- To push the prefix, click **Push Prefix** to open the DHCPv6 Push Prefix page. Choose the cluster or prefix template to which you want to push the prefix, then click **Push Prefix**.
  - To reclaim the prefix, click **Reclaim Prefix** to open the DHCPv6 Reclaim Prefix page. Choose the cluster or prefix template to which you want to reclaim the prefix, then click **Reclaim Prefix**.

## CLI Commands

Use **prefix name create ipv6address/length**. (The **prefix** command is a synonym for the **dhcp-prefix** command from previous releases.) Reload the DHCP server. For example:

```
nrcmd> prefix example-prefix create 2001:0db8::/32 [attribute=value]
nrcmd> dhcp reload
```

To apply a prefix template during prefix creation, use **prefix name create ipv6address/length template=name**. To apply a template to an existing prefix definition, use **prefix name applyTemplate template-name**. For example:

```
nrcmd> prefix example-prefix create 2001:0db8::/64 template=preftemp-1
nrcmd> prefix example-prefix applyTemplate template=preftemp-1
nrcmd> dhcp reload
```

You can set and enable the aforementioned attributes in the usual way. Add reservations by using **prefix name addReservation ipv6address/length lookup-key [-blob | -string]**. List leases by using **prefix name listLeases**. Manage DHCPv6 leases by using these commands:

```
nrcmd> lease6 {vpn-id/ | vpn-name/} ip6address[/prefix-length] activate
nrcmd> lease6 {vpn-id/ | vpn-name/} ip6address[/prefix-length] deactivate
nrcmd> lease6 {vpn-id/ | vpn-name/} ip6address[/prefix-length] force-available
```

```
nrcmd> lease6 {vpn-id/ | vpn-name/}ip6address[/prefix-length] get attribute
nrcmd> lease6 {vpn-id/ | vpn-name/}ip6address[/prefix-length] show
nrcmd> lease6 list
```

**Tip**

See the [“Reconfigure Support” section on page 26-32](#) for additional syntax.

You can get an exact count of the total prefixes and links for the DHCP server by using **dhcp getPrefixCount [vpn name | all]**. You can specify a VPN or all VPNs. Omitting the **vpn name** returns a count for the current VPN.

## Viewing Address Utilization for Prefixes


You can view the current address utilization for prefixes.

### Local Advanced and Regional Web UI

The function is available on the View Unified v6 Address Space page (see the [“Viewing Address Space” section on page 9-3](#)).

**Tip**

You can use the View Unified v6 Address Space page to push and reclaim prefixes. Click the **Push** or **Reclaim** link for the desired prefix. (See in the [“Creating and Editing Prefixes” section on page 26-24](#) for details.)

When you click the View icon () in the Current Usage column, or the Show Current Utilization for All Prefixes button, the View Current Prefix Utilization Report page appears.

**Note**

To ensure the proper subnet-to-server mapping on this page, you must update the regional address space view so that it is consistent with the relevant local cluster. Do this by pulling the replica address space, or reclaiming the subnet to push to the DHCP server. Also ensure that the particular DHCP server is running.

The other columns on the View Current Utilization Report page identify:

- **Range**—Address range of the prefix.
- **Type**—Whether the address space is a prefix or link.
- **Active Dynamic**—Addresses that are part of a dynamic range managed by DHCP and that are currently leased, but not reserved.

The Utilization Detail column items are expandable on the View Current Utilization Report page so that you can view the prefix or parent prefix data. Clicking the prefix or parent prefix name in this column opens the View Prefix Utilization Detail page.

The View Utilization Detail page is a read-only page that shows detailed address utilization attributes for the prefix or the parent prefix (identified as Totals). The address utilization attributes are described in [Table 26-5](#).

**Table 26-5 Address Utilization Attributes**

<b>Utilization Attribute</b>	<b>Description</b>
<i>aggregation-level</i>	Granularity of the utilization data. Prefix-level indicates the data is for the specific prefix; totals indicates the data is for the parent prefix, which is the sum of its prefix-level counters.
<i>dhcp-type</i>	DHCP address assignment type, which can be dhcp (stateful), stateless (option configuration), prefix-delegation, or infrastructure (maps a client address to a link without an address pool).
<b>Total Addresses</b>	
<i>active-dynamic</i>	Total number of dynamic leases in active use (leased, offered, released, expired, or revoked). The Active Dynamic category shows the states of these leases.
<i>total-reserved</i>	Total number of reserved leases.
<b>Active Dynamic</b>	
<i>offered</i>	Number of dynamic (unreserved) leases that are currently offered to clients, but not yet acknowledged as being leased.
<i>leased</i>	Number of dynamic leases that are currently acknowledged as leased to clients.
<i>expired</i>	Number of dynamic leases that are past the lease expiration period, but will not be available for other clients (except after the policy grace-period expires).
<i>revoked</i>	Number of dynamic leases that the client can no longer use, but that some other client could be using.
<b>Reserved</b>	
<i>reserved-active</i>	Number of reserved leases that clients are actively using.
<i>reserved-inactive</i>	Number of reserved leases that clients are not actively using.
<b>Unavailable</b>	
<i>unavail</i>	Number of unreserved dynamic leases that a client declines or the server marks with an address conflict (usually indicating configurations that need correcting).
<i>reserved-unavail</i>	Number of reserved leases that a client declines or the server marks with an address conflict (usually indicating configurations that need correcting).
<b>Deactivated</b>	
<i>deactivated</i>	Number of dynamic and reserved leases that clients are actively leasing (that are not offered, expired, or released), but that an administrator deactivated.
<i>leased-deactivated</i>	Number of dynamic leases that an administrator deactivated.
<i>reserved-leased-deactivated</i>	Number of reserved leases that an administrator deactivated.

## Viewing DHCPv6 Networks

To view the networks in the DHCPv6 address space, click **DHCPv6**, then **Networks** to open the View DHCPv6 Networks page. On this page you can add DHCPv6 links using a template and a template root prefix, as you would on the List/Add DHCPv6 Links page. Adding a link opens the Add DHCPv6 Link page. After creating the link, you can select it on the View DHCPv6 Networks page for editing.

## Editing DHCPv6 Server Attributes

You can edit DHCP server attributes related to DHCPv6. These attributes are:

- ***v6-client-class-lookup-id***—Expression that determines a client-class based on the DHCPv6 client request and returns a string or <none>. The expression must return a string that is the name of a configured client-class. The attribute has no preset value.
- ***max-client-leases***—Maximum number of leases a DHCPv6 client can have on a link. Do not use this attribute to limit clients to one lease only. The preset is 50.

### Local Basic or Advanced Web UI

Click **DHCPv6**, then **DHCP Server** to open the Manage DHCP Server page. Click the Local DHCP Server link to open the Edit DHCP Server page, modify the aforementioned DHCPv6 attribute values, then click **Modify Server**.

### CLI Commands

Use **dhcp** to show the aforementioned DHCPv6 server attributes, then modify them by using **dhcp set**.

## Configuring DHCPv6 Policies

You can edit DHCPv6 policy attributes, which are:

- ***affinity-period***—See the [“Lease Affinity” section on page 26-6](#) (no preset value).
- ***allow-non-temporary-addresses***—Enable or disable DHCPv6 clients requesting nontemporary (IA\_NA) addresses (preset value enable).
- ***allow-rapid-commit***—With Rapid Commit enabled, clients receive information (when solicited) on committed addresses, which are then more quickly committed with a client request (preset value disable). Use Rapid Commit only if one DHCP server is servicing clients, otherwise it might seem like the client is receiving multiple addresses. (See the [“DHCPv6 Policy Hierarchy” section on page 26-9](#) for special handling of this attribute, and Reconfigure support, when used in an embedded or named policy for a prefix.)
- ***allow-temporary-addresses***—Enable or disable DHCPv6 clients requesting temporary (IA\_IA) addresses (preset value enable).
- ***default-prefix-length***—For prefix delegation, default prefix length of the delegated prefix if the client or router does not explicitly request it (or *allow-client-hints* is disabled); must always be less than or equal to the prefix range prefix length (preset value 64 bytes).
- ***preferred-lifetime***—Default and maximum preferred lifetime for leases (preset value 1 week).

- **v6-reply-options**—DHCPv6 options returned in replies to clients (no preset value). (See the “[DHCPv6 Policy Hierarchy](#)” section on page 26-9 for special handling of this attribute when used in an embedded or named policy for a prefix.)
- **valid-lifetime**—Default and maximum valid lifetime for leases (preset value 2 weeks).

**Tip**

For details on the Reconfigure attributes, see the “[Reconfigure Support](#)” section on page 26-32.

## Local Advanced Web UI

From the **DHCPv6** menu, choose **Policies** to open the List/Add DHCP Policies page. Click **Add Policy** to add a new policy on the Add DHCP Policy page or click an existing policy to open the Edit DHCP Policy page. Both pages have DHCPv4 and DHCPv6 options sections. Add (or delete) options and set attributes as desired, then click **Add Policy** or **Modify Policy**.

## CLI Commands

Use **policy list** or **policy name show** to show the aforementioned policy attributes, then modify them by using **policy name set** or **enable**.

# Configuring DHCPv6 Client-Classes

You can configure DHCPv6 client-class attributes, which are:

- **v6-client-lookup-id**—Key value to use to look up the DHCPv6 client in the client database (locally or through LDAP), specified as an expression that evaluates to a string (or a blob as a valid string).
- **v6-override-client-id**—Value that replaces any client-identity value in an incoming packet, specified as an expression that evaluates to a blob.

## Local Advanced Web UI

- Step 1** From the **DHCPv6** menu, choose **Client-Classes** to open the List/Add DHCP Client Classes page.
- Step 2** Click an existing client-class to open the Edit DHCP Client-Class page, or click **Add Client-Class** to add a new client-class on the Add DHCP Client-Class page. Both pages include the aforementioned attributes.
- Step 3** Click **Modify Client-Class**.
- Step 4** To generate clients, be sure that *validate-client-name-as-mac* is disabled for the DHCP server. This attribute appears on the Edit DHCP Server page under the Client-Class attributes.
- Step 5** Reload the DHCP server.

## CLI Commands

Use **client-class list** or **client-class name show** to show the aforementioned client-class attributes, then modify them using **client-class name set**. To generate clients, be sure that *validate-client-name-as-mac* is disabled for the DHCP server.

## Configuring DHCPv6 Clients

You can configure DHCPv6 clients.

### Local Advanced Web UI

From the **DHCP v6** menu, choose **Clients** to open the List/Add DHCP Clients page. Click an existing client to open the Edit DHCP Client page or click **Add Client** to add a new client-class on the List/Add DHCP Client page, choose the client-class that includes the DHCPv6 attributes that were set (see the [“Configuring DHCPv6 Client-Classes”](#) section on page 26-30), then click **Modify Client**.



Tip

Disable the *validate-client-name-as-mac* attribute for the DHCP server.

### CLI Commands

Use **client list** or **client name show** to show the existing clients. To set the client-class name for the client, use **client name set client-class-name=value**. Also ensure that the *validate-client-name-as-mac* attribute is disabled for the DHCP server.

## Setting DHCPv6 Options

Set DHCPv6 options and vendor options when you create or edit policies (embedded or named) for prefixes. (See the [“DHCPv6 Policy Hierarchy”](#) section on page 26-9 for special handling of the *v6-options* and *v6-vendor-options* policy attributes when used in an embedded or named policy on a prefix.)

### Local Advanced Web UI

The DHCPv6 options coexist along with the DHCPv4 options on the Add DHCP Policy or Edit DHCP Policy page. Note that the vendor options appear only if you create these options (see the [“Creating DHCP Option Definition Sets and Option Definitions”](#) section on page 21-8).

You can select the options from the drop-down lists. If option descriptions exist, they appear under the Name and Number headings, which you can click to sort the entries.

### CLI Commands

Use **policy name setV6Option** or **policy name setV6VendorOption**. The option settings require an option name (or ID) and a value. For example:

```
nrcmd> policy dhcpv6-policy setV6Option dns-servers 2222::1,2222::2
nrcmd> policy dhcpv6-policy setV6VendorOption 1234 2222::3,2222::4
```



## Reconfigure Support

For DHCPv6, a server can send a DHCPRECONFIGURE message to a client to inform the client that the server has new or updated configuration parameters. If so authorized and through proper authentication, the client then immediately initiates a Renew, Rebind, or Information-request reply transaction with the server so that the client can retrieve the new data. Without this support, a client must wait until it renews its lease to get configuration updates.

You can have the server unicast the Reconfigure packet or deliver it through a relay agent. If you do not specify either way, the client's client-class policy, requested lease's prefix or link policies, or system\_default\_policy (but not the client policy) determines the preferred method. If the unicast method is not available (the client has no valid address lease), the server uses the relay agent; with no relay agent, the server tries to unicast; failing both results in an error. With the unicast method, if the specified lease is not usable, the server selects the lease with the longest valid lifetime.

The server and client negotiate Reconfigure support through the added security of a reconfigure key. The internal process is basically:

1. The client sends the server a REQUEST, SOLICIT, or ADVERTISE packet that includes the *reconfigure-accept* option (20) to indicate that the client wants to accept Reconfigure messages. (Conversely, the DHCP server can send a *reconfigure-accept* option to the client about whether the client should accept Reconfigure messages.) This option is required for Reconfigure support.
2. If the Cisco Network Registrar policy for the client has the *reconfigure* attribute set to **allow** or **require** (rather than **disallow**), the DHCP server accepts the packet and generates a reconfigure key for the client. (The server records the key value and its generation time in the *client-reconfigure-key* and *client-reconfigure-key-generation-time* attributes for the DHCPv6 lease.)
3. The server sends a Reply packet to the client with the reconfigure key in the *auth* option (11) along with the *reconfigure-accept* option.
4. The client records the reconfigure key to authenticate Reconfigure messages from the server.
5. When the server wants to reconfigure the client, it sends a Reconfigure packet with the *reconfigure-message* option (19) and an *auth* option containing a hash generated from the packet and the reconfigure key. The *reconfigure-message* option indicates in the *msg-type* field whether the client should respond with a Renew or an Information-request packet.
6. Upon receiving the packet, the client validates that the *auth* option contains the valid hash, then returns a Renew, Rebind, or Information-request packet. This packet includes an Option Request (*oro*) option (6) to indicate specific option updates. (If the server does not receive a reply from the client in a preconfigured timeout value of 2 seconds, the server retransmits the Reconfigure message at most 8 times, then aborts the reconfigure process for the client.)
7. The server sends the client a Reply packet that includes options for configuration parameters. The packet might also include options containing addresses and new values for other configuration parameters, even if the client did not request them. The client records these changes.

**Note**

For details on how Reconfigure support affects particular DHCP extension points, see the [“Extension Dictionaries” section on page 29-23](#).

### Local Advanced Web UI

The List DHCP Leases for Prefix page includes a **Reconfigure** button in the Reconfigure column for each lease so that you can initiate a reconfiguration request for that particular lease.



## CLI Commands

To support Reconfigure, Cisco Network Registrar includes the following syntax for the **lease6** command:

```
lease6 ipaddr reconfigure [renew | rebind | information-request] [-unicast | -via-relay]
```

The options determine whether to have the client respond to the Reconfigure message with a Renew, Rebind, or Information-request packet, and whether the server should unicast or go through a relay agent. The **lease6 list** and **show** commands also display values for these related attributes:

- **client-reconfigure-key**—128-bit key that the server generates for Reconfigure messages to the client.
- **client-reconfigure-key-generation-time**—Time at which the server generated the *client-reconfigure-key*.

The **policy** command includes two related attribute settings:

- **reconfigure**—Whether to allow (1), disallow (2), or require (3) Reconfigure support; the preset value is allow (1).
- **reconfigure-via-relay**—Whether to allow reconfiguration over a relay agent; the preset value is false, whereby reconfiguration notification is by unicasting from the server.

## DNS Update for DHCPv6

For details on enabling and configuring DNS update for DHCPv6 clients, see the [“DNS Update for DHCPv6” section on page 28-2](#).

