CHAPTER 28

Configuring DNS Update

The DNS Update protocol (RFC 2136) integrates DNS with DHCP. The latter two protocols are complementary; DHCP centralizes and automates IP address allocation, while DNS automatically records the association between assigned addresses and hostnames. When you use DHCP with DNS update, this configures a host automatically for network access whenever it attaches to the IP network. You can locate and reach the host using its unique DNS hostname. Mobile hosts, for example, can move freely without user or administrator intervention.

This chapter explains how to use DNS update with Cisco Network Registrar servers, and its special relevance to Windows client systems.

See Also

DNS Update Process

To configure DNS updates, you must:

1. Create a DNS update configuration for a forward or reverse zone or both. See the "Creating DNS Update Configurations" section on page 28-5.
2. Use this DNS update configuration in either of two ways:
   - Specify the DNS update configuration on a named, embedded, or default DHCP policy. See the "Creating and Applying DHCP Policies" section on page 21-3.
   - Define a DNS update map to autoconfigure a single DNS update relationship between a Cisco Network Registrar DHCP server or failover pair and a DNS server or High-Availability (HA) pair. Specify the update configuration in the DNS update map. See the "Creating DNS Update Maps" section on page 28-7.

3. Optionally define access control lists (ACLs) or transaction signatures (TSIGs) for the DNS update. See the "Configuring Access Control Lists and Transaction Security" section on page 28-8.
4. Optionally create one or more DNS update policies based on these ACLs or TSIGs and apply them to the zones. See the "Configuring DNS Update Policies" section on page 28-12.
5. Adjust the DNS update configuration for Windows clients, if necessary; for example, for dual zone updates. See the "Configuring DNS Update for Windows Clients" section on page 28-18.
6. Configure DHCP clients to supply hostnames or request that Cisco Network Registrar generate them.
7. Reload the DHCP and DNS servers, if necessary based on the edit mode.

Special DNS Update Considerations

Consider these two issues when configuring DNS updates:

- For security purposes, the Cisco Network Registrar DNS update process does not modify or delete a name an administrator manually enters in the DNS database.
- If you enable DNS update for large deployments, and you are not using HA DNS (see Chapter 18, "Configuring High-Availability DNS Servers"), divide primary DNS and DHCP servers across multiple clusters. DNS update generates an additional load on the servers.

DNS Update for DHCPv6

Cisco Network Registrar currently supports DHCPv6 DNS update over IPv4 only. For DHCPv6, DNS update applies to nontemporary stateful addresses only, not delegated prefixes.

DNS update for DHCPv6 involves AAAA and PTR RR mappings for leases. Cisco Network Registrar 7.2 supports server- or extension-synthesizing fully qualified domain names and the DHCPv6 *client-fqdn* option (39).

Because Cisco Network Registrar is compliant with RFCs 4701, 4703, and 4704, it supports the DHCID resource record (RR). All RFC-4703-compliant updaters can generate DHCID RRs and result in data that is a hash of the client identifier (DUID) and the FQDN (per RFC 4701). Nevertheless, you can use AAAA and DHCID RRs in update policy rules.

DNS update processing for DHCPv6 is similar to that for DHCPv4 except that a single FQDN can have more than one lease, resulting in multiple AAAA and PTR RRs for a single client. The multiple AAAA RRs can be under the same name or a different name; however, PTR RRs are always under a different name, based on the lease address. RFC-4703-compliant updaters use the DHCID RR to avoid collisions among multiple clients.

Note Because DHCPv4 uses TXT RRs and DHCPv6 uses DHCID RRs for DNS update, to avoid conflicts, dual-stack clients cannot use single forward FQDNs. These conflicts primarily apply to client-requested names and not generated names, which are generally unique. To avoid these conflicts, use different zones for the DHCPv4 and DHCPv6 names.

Note If the DNS server is down and the DHCP server can not complete the DNS updates to remove RRs added for a DHCPv6 lease, the lease continues to exist in the AVAILABLE state. Only the same client reuses the lease.

See Also

DHCPv6 Upgrade Considerations
Generating Synthetic Names in DHCPv6
Determining Reverse Zones for DNS Updates, page 28-4
Using the Client FQDN, page 28-4

DHCPv6 Upgrade Considerations

If you use any policy configured prior to Cisco Network Registrar 7.2 that references a DNS update object for DHCPv6 processing (see the "DHCPv6 Policy Hierarchy" section on page 26-9), after the upgrade, the server begins queuing DNS updates to the specified DNS server or servers. This means that DNS updates might automatically (and unexpectedly) start for DHCPv6 leases.

Caution If you use earlier versions of Cisco Network Registrar or other DNS servers, you might experience interoperability issues for zone transfers and DNS updates, because of recent DHCID RR standards changes. You might need to upgrade DNS servers to support DHCPv6 DNS updates.

Generating Synthetic Names in DHCPv6

If clients do not supply hostnames, DHCPv6 includes a synthetic name generator. Because a DHCPv6 client can have multiple leases, Cisco Network Registrar uses a different mechanism than that for DHCPv4 to generate unique hostnames. The *v6-synthetic-name-generator* attribute for the DNS update configuration allows appending a generated name to the *synthetic-name-stem* based on the:

- Hash of the client DHCP Unique Identifier (DUID) value (the preset value).
- Raw client DUID value (as a hex string with no separators).
- CableLabs *cablelabs-17* option *device-id* suboption value (as a hex string with no separators, or the hash of the client DUID if not found).
- CableLabs *cablelabs-17* option *cm-mac-address* suboption value (as a hex string with no separators, or the hash of the client DUID if not found).

Caution Some generation methods might cause privacy issues if the domain is accessible from the Internet.

See the "Creating DNS Update Configurations" section on page 28-5 for how to create a DNS update configuration with synthetic name generation.

In the CLI, an example of this setting is:

```
nrcmd> dhcp-dns-update example-update-config set v6-synthetic-name-generator=hashed-duid
```

Determining Reverse Zones for DNS Updates

The DNS update configuration uses the prefix length value in the specified *reverse-zone-prefix-length* attribute to generate a reverse zone in the ip6.arpa domain. You do not need to specify the full reverse zone, because you can synthesize it by using the ip6.arpa domain. You set this attribute for the reverse DNS update configuration (see the "Creating DNS Update Configurations" section on page 28-5). Here are some rules for *reverse-zone-prefix-length*:

- 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 therein.
- 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 (see the "Configuring Prefixes" section on page 26-18).

Note that to synthesize the reverse zone name, the *synthesize-reverse-zone* attribute must remain enabled for the DHCP server. Thus, the order in which a reverse zone name is synthesized for DHCPv6 is:

1. Use the full *reverse-zone-name* in the reverse DNS update configuration.
2. Base it on the ip6.arpa zone from the *reverse-zone-prefix-length* in the reverse DNS update configuration.
3. Base it on the ip6.arpa zone from the *reverse-zone-prefix-length* in the prefix definition.
4. Base it on the ip6.arpa zone from the prefix length for the *address* in the prefix definition.

In the CLI, an example of setting the reverse zone prefix length is:

```
nrcmd> dhcp-dns-update example-update-config set reverse-zone-prefix-length=32
```

To create a reverse zone for a prefix in the web UI, the List/Add Prefixes page includes a **Create Reverse Zone** button for each prefix. (See the "Creating and Editing Prefixes" section on page 26-24.)

The CLI also provides the **prefix** *name* **createReverseZone** [**–range**] command to create a reverse zone for a prefix (from its address or range value). Delete the reverse zone by using **prefix** *name* **deleteReverseZone** [**–range**].

You can also create a reverse zone from a DHCPv4 subnet or DHCPv6 prefix by entering the subnet or prefix value when directly configuring the reverse zone. See the "Adding Primary Reverse Zones" section on page 15-12 for details.

Using the Client FQDN

The existing DHCP server *use-client-fqdn* attribute controls whether the server pays attention to the DHCPv6 client FQDN option in the request. The rules that the server uses to determine which name to return when multiple names exist for a client are in the following order of preference:

1. The server FQDN that uses the client requested FQDN if it is in use for any lease (even if not considered to be in DNS).
2. The FQDN with the longest valid lifetime considered to be in DNS.
3. The FQDN with the longest valid lifetime that is not yet considered to be in DNS.

Creating DNS Update Configurations

A DNS update configuration defines the DHCP server framework for DNS updates to a DNS server or HA DNS server pair. It determines if you want to generate forward or reverse zone DNS updates (or both). It optionally sets TSIG keys for the transaction, attributes to control the style of autogenerated hostnames, and the specific forward or reverse zone to be updated. You must specify a DNS update configuration for each unique server relationship.

For example, if all updates from the DHCP server are directed to a single DNS server, you can create a single DNS update configuration that is set on the server default policy. To assign each group of clients in a client-class to a corresponding forward zone, set the forward zone name for each in a more specific client-class policy.

Local Advanced and Regional Web UI

Step 1 From the **DHCP** menu, choose **DNS Updates** to open the List/Add DNS Update Configurations page.

Step 2 Click **Add DNS Update Configuration** to open the Add DNS Update Configuration page.

Step 3 Enter a name for the update configuration in the *name* attribute field.

Step 4 Click the appropriate *dynamic-dns* setting:

- **update-none**—Do not update forward or reverse zones.
- **update-all**—Update forward and reverse zones (the default value).
- **update-fwd-only**—Update forward zones only.
- **update-reverse-only**—Update reverse zones only.

Step 5 Set the other attributes appropriately:

a. If necessary, enable *synthesize-name* and set the *synthetic-name-stem* value.

You can set the stem of the default hostname to use if clients do not supply hostnames, by using *synthetic-name-stem*. For DHCPv4, enable the *synthesize-name* attribute to trigger the DHCP server to synthesize unique names for clients based on the value of the *synthetic-name-stem*. The resulting name is the name stem appended with the hyphenated IP address. For example, if you specify a *synthetic-name-stem* of **host** for address 192.168.50.1 in the example.com domain, and enable the *synthesize-name* attribute, the resulting hostname is host-192-168-50-1.example.com. The preset value for the synthetic name stem is **dhcp**.

The *synthetic-name-stem* must:

- Be a relative name without a trailing dot.
- Include alphanumeric values and hyphens (**–**) only. Space characters and underscores become hyphens and other characters are removed.
- Include no leading or trailing hyphen characters.
- Have DNS hostnames of no more than 63 characters per label and 255 characters in their entirety. The algorithm uses the configured forward zone name to determine the number of available characters for the hostname, and truncates the end of the last label if necessary.

For DHCPv6, see the "Generating Synthetic Names in DHCPv6" section on page 28-3.

b. Set *forward-zone-name* to the forward zone, if updating forward zones. Note that the policy *forward-zone-name* takes precedence over the one set in the DNS update configuration.

For DHCPv6, the server ignores the client and client-class policies when searching for a *forward-zone-name* value in the policy hierarchy. The search for a forward zone name begins with the prefix embedded policy.

c. For DHCPv4, set *reverse-zone-name* to the reverse (in.addr.arpa) zone to be updated with PTR and TXT records. If unset and the DHCP server *synthesize-reverse-zone* attribute is enabled, the server synthesizes a reverse zone name based on the address of each lease, scope subnet number, and DNS update configuration (or scope) *dns-host-bytes* attribute value.

   The *dns-host-bytes* value controls the split between the host and zone parts of the reverse zone name. The value sets the number of bytes from the lease IP address to use for the hostname; the remaining bytes are used for the in-addr.arpa zone name. A value of 1 means use just one byte for the host part of the domain and the other three from the domain name (reversed). A value of 4 means use all four bytes for the host part of the address, thus using just the in-addr.arpa part of the domain. If unset, the server synthesizes an appropriate value based on the scope subnet size, or if the *reverse-zone-name* is defined, calculates the host bytes from this name.

   For DHCPv6, see the "Determining Reverse Zones for DNS Updates" section on page 28-4.

d. Set *server-addr* to the IP address of the primary DNS server for the forward zone (or reverse zone if updating reverse zones only.

e. Set *server-key* and *backup-server-key* if you are using a TSIG key to process all DNS updates (see the "Transaction Security" section on page 28-9).

f. Set *backup-server-addr* to the IP address of the backup DNS server, if HA DNS is configured.

g. If necessary, enable or disable *update-dns-first* (preset value disabled) or *update-dns-for-bootp* (preset value enabled). The *update-dns-first* setting controls whether DHCP updates DNS before granting a lease. Enabling this attribute is not recommended.

Step 6 At the regional level, you can also push update configurations to the local clusters, or pull them from the replica database on the List/Add DNS Update Configurations page.

Step 7 Click **Add DNS Update Configuration**.

Step 8 To specify this DNS update configuration on a policy, see the "Creating and Applying DHCP Policies" section on page 21-3.

CLI Commands

Use **dhcp-dns-update** *name* **create**. For example:

```
nrcmd> dhcp-dns-update example-update-config create
```

Set the *dynamic-dns* attribute to its appropriate value (update-none, update-all, update-fwd-only, or update-reverse-only). For example:

```
nrcmd> dhcp-dns-update example-update-config set dynamic-dns=update-all
```

See Also

DNS Update Process, page 28-1
Special DNS Update Considerations, page 28-2
DNS Update for DHCPv6, page 28-2

Creating DNS Update Maps

A DNS update map facilitates configuring DNS updates so that the update properties are synchronized between HA DNS server pairs or DHCP failover server pairs, based on an update configuration, so as to reduce redundant data entry. The update map applies to all the primary zones that the DNS pairs service, or all the scopes that the DHCP pairs service. You must specify a policy for the update map. To use this function, you must be an administrator assigned the server-management subrole of the dns-management or central-dns-management role, and the dhcp-management role (for update configurations).

Local Advanced and Regional Web UI

Step 1 From the **DNS** menu, choose **Update Maps** to open the List/Add DNS Update Maps page.

Step 2 Click **Add DNS Update Map** to open the Add DNS Update Map page.

Step 3 Enter a name for the update map in the Name field.

Step 4 Enter the DNS update configuration from the previous section in the *dns-config* field.

Step 5 Set the kind of policy selection you want for the *dhcp-policy-selector* attribute. The choices are:

- **use-named-policy**—Use the named policy set for the *dhcp-named-policy* attribute (the preset value).
- **use-client-class-embedded-policy**—Use the embedded policy from the client-class set for the *dhcp-client-class* attribute.
- **use-scope-embedded-policy**—Use the embedded policy from the scope.

Step 6 If using update ACLs (see the "Configuring Access Control Lists and Transaction Security" section on page 28-8) or DNS update policies (see the "Configuring DNS Update Policies" section on page 28-12), set either the *dns-update-acl* or *dns-update-policy-list* attribute. Either value can be one or more addresses separated by commas. The *dns-update-acl* takes precedence over the *dns-update-policy-list*.

If you omit both values, a simple update ACL is constructed whereby only the specified DHCP servers or failover pair can perform updates, along with any *server-key* value set in the update configuration specified for the *dns-config* attribute.

Step 7 Click **Add DNS Update Map**.

Step 8 At the regional level, you can also push update maps to the local clusters, or pull them from the replica database on the List/Add DNS Update Maps page.

CLI Commands

Specify the name, cluster of the DHCP and DNS servers (or DHCP failover or HA DNS server pair), and the DNS update configuration when you create the update map, using **dns-update-map** *name* **create** *dhcp-cluster dns-cluster dns-config*. For example:

```
nrcmd> dns-update-map example-update-map create Example-cluster Boston-cluster
    example-update-config
```

Set the *dhcp-policy-selector* attribute value to use-named-policy, use-client-class-embedded-policy, or use-scope-embedded-policy. If using the use-named-policy value, also set the *dhcp-named-policy* attribute value. For example:

```
nrcmd> dns-update-map example-update-map set dhcp-policy-selector=use-named-policy
nrcmd> dns-update-map example-update-map set dhcp-named-policy=example-policy
```

Configuring Access Control Lists and Transaction Security

ACLs are authorization lists, while transaction signatures (TSIG) is an authentication mechanism:

- ACLs enable the server to allow or disallow the request or action defined in a packet.
- TSIG ensures that DNS messages come from a trusted source and are not tampered with.

For each DNS query, update, or zone transfer that is to be secured, you must set up an ACL to provide permission control. TSIG processing is performed only on messages that contain TSIG information. A message that does not contain, or is stripped of, this information bypasses the authentication process.

For a totally secure solution, messages should be authorized by the same authentication key. For example, if the DHCP server is configured to use TSIG for DNS updates and the same TSIG key is included in the ACL for the zones to be updated, then any packet that does not contain TSIG information fails the authorization step. This secures the update transactions and ensures that messages are both authenticated and authorized before making zone changes.

ACLs and TSIG play a role in setting up DNS update policies for the server or zones, as described in the "Configuring DNS Update Policies" section on page 28-12.

See Also

Access Control Lists
Configuring Zones for Access Control Lists, page 28-9
Transaction Security, page 28-9

Access Control Lists

You assign ACLs on the DNS server or zone level. ACLs can include one or more of these elements:

- **IP address**—In dotted decimal notation; for example, 192.168.1.2.
- **Network address**—In dotted decimal and slash notation; for example, 192.168.0.0/24. In this example, only hosts on that network can update the DNS server.
- **Another ACL**—Must be predefined. You cannot delete an ACL that is embedded in another one until you remove the embedded relationship. You should not delete an ACL until all references to that ACL are deleted.
- **Transaction Signature (TSIG) key**—The value must be in the form **key** *value*, with the keyword **key** followed by the secret value. To accommodate space characters, the entire list must be enclosed in double quotes. For TSIG keys, see the "Transaction Security" section on page 28-9.

You assign each ACL a unique name. However, the following ACL names have special meanings and you cannot use them for regular ACL names:

- **any**—Anyone can perform a certain action
- **none**—No one can perform a certain action
- **localhost**—Any of the local host addresses can perform a certain action
- **localnets**—Any of the local networks can perform a certain action

Note the following:

- If an ACL is not configured, **any** is assumed.
- If an ACL is configured, at least one clause must allow traffic.

- The negation operator (**!**) disallows traffic for the object it precedes, but it does not intrinsically allow anything else unless you also explicitly specify it. For example, to disallow traffic for the IP address 192.168.50.0 only, use **!192.168.50.0, any**.

Local Advanced Web UI

Click **DNS**, then **ACLs** to open the List/Add Access Control Lists page. Add an ACL name and match list. Note that a **key** *value* pair should not be in quotes. At the regional level, you can additionally pull replica ACLs or push ACLs to local clusters.

CLI Commands

Use **acl** *name* **create** *match-list*, which takes a name and one or more ACL elements. The ACL list is comma-separated, with double quotes surrounding it if there is a space character. The CLI does not provide the pull/push function.

For example, the following commands create three ACLs. The first is a key with a value, the second is for a network, and the third points to the first ACL. Including an exclamation point (**!**) before a value negates that value, so that you can exclude it in a series of values:

```
nrcmd> acl sec-acl create "key h-a.h-b.example.com."
nrcmd> acl dyn-update-acl create "!192.168.2.13,192.168.2.0/24"
nrcmd> acl main-acl create sec-acl
```

Configuring Zones for Access Control Lists

To configure ACLs for the DNS server or zones, set up a DNS update policy, then define this update policy for the zone (see the "Configuring DNS Update Policies" section on page 28-12).

Transaction Security

Transaction Signature (TSIG) RRs enable the DNS server to authenticate each message that it receives, containing a TSIG. Communication between servers is not encrypted but it becomes authenticated, which allows validation of the authenticity of the data and the source of the packet.

When you configure the Cisco Network Registrar DHCP server to use TSIG for DNS updates, the server appends a TSIG RR to the messages. Part of the TSIG record is a message authentication code.

When the DNS server receives a message, it looks for the TSIG record. If it finds one, it first verifies that the key name in it is one of the keys it recognizes. It then verifies that the time stamp in the update is reasonable (to help fight against traffic replay attacks). Finally, the server looks up the key shared secret that was sent in the packet and calculates its own authentication code. If the resulting calculated authentication code matches the one included in the packet, then the contents are considered to be authentic.

See Also

Creating TSIG Keys
Generating Keys, page 28-10
Considerations for Managing Keys, page 28-11
Adding Supporting TSIG Attributes, page 28-11

Creating TSIG Keys

Note If you want to enable key authentication for Address-to-User Lookup (ATUL) support, you must also define a key identifier (*id* attribute value). See the "Setting DHCP Forwarding" section on page 24-24.

Local Advanced Web UI

From the **Administration** menu or the **DNS** menu, choose **Keys**, to open the List/Add Encryption Keys page.

For a description of the Algorithm, Security Type, Time Skew, Key ID, and Secret values, see Table 28-1 on page 28-10. See also the "Considerations for Managing Keys" section on page 28-11.

To edit a TSIG key, click its name on the List/Add Encryption Keys page to open the Edit Encryption Key page.

At the regional level, you can additionally pull replica keys, or push keys to local clusters.

CLI Commands

Use **key** *name* **create** *secret*. Provide a name for the key (in domain name format; for example, hosta-hostb-example.com.) and a minimum of the shared secret as a base-64 encoded string (see Table 28-1 on page 28-10 for a description of the optional time skew attribute). An example in the CLI would be:

```
nrcmd> key hosta-hostb-example.com. create secret-string
```

Generating Keys

It is recommended that you use the Cisco Network Registrar **cnr_keygen** utility to generate TSIG keys so that you add them or import them using **import keys**.

Execute the **cnr_keygen** key generator utility from a DOS prompt, or a Solaris or Linux shell:

- On Windows, the utility is in the *install-path*\bin folder.
- On Solaris and Linux, the utility is in the *install-path*/usrbin directory.

An example of its usage (on Solaris and Linux) is:

```
> /opt/nwreg2/local/usrbin/cnr_keygen -n a.b.example.com. -a hmac-md5 -t TSIG -b 16
    -s 300
        key "a.b.example.com." {
        algorithm hmac-md5;
        secret "xGVCsFZ0/6e0N97HGF50eg==";
        # cnr-time-skew 300;
        # cnr-security-type TSIG;
};
```

The only required input is the key name. The options are described in Table 28-1.

Table 28-1 Options for the cnr_keygen Utility

Option	Description
–n *name*	Key name. Required. The maximum length is 255 bytes.
–a hmac-md5	Algorithm. Optional. Only hmac-md5 is currently supported.

Table 28-1 Options for the cnr_keygen Utility (continued)

Option	Description
–b *bytes*	Byte size of the secret. Optional. The preset value is 16 bytes. The valid range is 1 through 64 bytes.
–s *skew*	Time skew for the key, in seconds. This is the maximum difference between the time stamp in packets signed with this key and the local system time. Optional. The preset value is 5 minutes. The range is one second through one hour.
–t tsig	Type of security used. Optional. Only TSIG is currently supported.
–h	Help. Optional. Displays the syntax and options of the utility.
–v	Version. Optional. Displays the version of the utility.

The resulting secret is base64-encoded as a random string.

You can also redirect the output to a file if you use the right-arrow (**>**) or double-right-arrow (**>>**) indicators at the end of the command line. The > writes or overwrites a given file, while the **>>** appends to an existing file. For example:

```
> /opt/nwreg2/local/usrbin/cnr_keygen -n example.com > keyfile.txt
> /opt/nwreg2/local/usrbin/cnr_keygen -n example.com >> addtokeyfile.txt
```

You can then import the key file into Cisco Network Registrar using the CLI to generate the keys in the file. The key import can generate as many keys as it finds in the import file. The path to the file should be fully qualified. For example:

```
nrcmd> import keys keydir/keyfile.txt
```

Considerations for Managing Keys

If you generate your own keys, you must enter them as a base64-encoded string (See RFC 4648 for more information on base64 encoding). This means that the only characters allowed are those in the base64 alphabet and the equals sign (=) as pad character. Entering a nonbase64-encoded string results in an error message.

Here are some other suggestions:

- Do not add or modify keys using batch commands.
- Change shared secrets frequently; every two months is recommended. Note that Cisco Network Registrar does not explicitly enforce this.
- The shared secret length should be at least as long as the keyed message digest (HMAC-MD5 is 16 bytes). Note that Cisco Network Registrar does not explicitly enforce this and only checks that the shared secret is a valid base64-encoded string, but it is the policy recommended by RFC 2845.

Adding Supporting TSIG Attributes

To add TSIG support for a DNS update configuration (see the "Creating DNS Update Configurations" section on page 28-5), set these attributes:

- *server-key*
- *backup-server-key*

Configuring DNS Update Policies

DNS update policies provide a mechanism for managing update authorization at the RR level. Using update policies, you can grant or deny DNS updates based on rules that are based on ACLs as well as RR names and types. ACLs are described in the "Access Control Lists" section on page 28-8.

See Also

Compatibility with Previous Cisco Network Registrar Releases
Creating and Editing Update Policies, page 28-12
Defining and Applying Rules for Update Policies, page 28-13

Compatibility with Previous Cisco Network Registrar Releases

Previous Cisco Network Registrar releases used static RRs that administrators entered, but that DNS updates could not modify. This distinction between static and dynamic RRs no longer exists. RRs can now be marked as protected or unprotected (see the "Protecting Resource Record Sets" section on page 16-3). Administrators creating or modifying RRs can now specify whether RRs should be protected. A DNS update cannot modify a protected RR set, even if an RR of the given type does not yet exist in the set.

Note Previous releases allowed DNS updates only to A, TXT, PTR, CNAME and SRV records. This was changed to allow updates to all but SOA and NS records in unprotected name sets. To remain compatible with a previous release, use an update policy to limit RR updates.

Creating and Editing Update Policies

Creating an update policy initially involves creating a name for it.

Local Advanced Web UI

Step 1 From the **DNS** menu, choose **Update Policies** to open the List DNS Update Policies page.

Step 2 Click **Add Policy** to open the Add DNS Update Policy page.

Step 3 Enter a name for the update policy.

Step 4 Proceed to the "Defining and Applying Rules for Update Policies" section.

CLI Commands

Use **update-policy** *name* **create**; for example:

```
nrcmd> update-policy policy1 create
```

Defining and Applying Rules for Update Policies

DNS update policies are effective only if you define rules for each that grant or deny updates for certain RRs based on an ACL. If no rule is satisfied, the default (last implicit) rule is to deny all updates (**"deny any wildcard * *"**).

See Also

Defining Rules for Named Update Policies
Applying Update Policies to Zones, page 28-15

Defining Rules for Named Update Policies

Defining rules for named update policies involves a series of Grant and Deny statements.

Local Advanced Web UI

Step 1 Create an update policy, as described in the "Creating and Editing Update Policies" section on page 28-12, or edit it.

Step 2 On the Add DNS Update Policies or Edit DNS Update Policy page:

a. Enter an optional value in the Index field.

b. Click Grant to grant the rule, or Deny to deny the rule.

c. Enter an access control list in the ACL List field.

d. Choose a keyword from the Keyword drop-down list.

e. Enter a value based on the keyword in the Value field. This can be a RR or subdomain name, or, if the **wildcard** keyword is used, it can contain wildcards (see Table 28-2).

Table 28-2 Wildcard Values for Update Policy Rules

Wildcard	Description
*	Matches zero or more characters. For example, the pattern **example*** matches all strings starting with *example*, including **example–**.
?	Matches a single character only. For example, the pattern **example?.com** matches **example1.com** and **example2.com**, but not **example.com**.
/[/]	Matches any characters in the (escaped) brackets; for example, **/[abc/]**. Each square bracket must be escaped using a slash (/). The characters can also be in a range; for example, **/[0–9/]** and **/[a–z/]**. If a pattern should include a hyphen, make the hyphen the first character; for example, **example/[–a–z/]**.

f. Enter one or more RR types, separated by commas, in the RR Types field, or use * for "all RRs." You can use negated values, which are values prefixed by an exclamation point; for example, **!PTR**.

g. Click **Add Policy**.

Step 3 At the regional level, you can also push update policies to the local clusters, or pull them from the replica database on the List DNS Update Policies page.

Step 4 To edit an update policy, click the name of the update policy on the List DNS Update Policies page to open the Edit DNS Update Policy page, make changes to the fields, then click **Edit Policy**.

CLI Commands

Create or edit an update policy (see the "Creating and Editing Update Policies" section on page 28-12, then use **update-policy** *name* **rules add** *rule*, with *rule* being the rule. (See Table 28-2 on page 28-13 for the rule wildcard values.) For example:

```
nrcmd> update-policy policy1 rules add "grant 192.168.50.101 name host1 A,TXT" 0
```

The rule is enclosed in quotes. To parse the rule syntax for the example:

- **grant**—Action that the server should take, either **grant** or **deny**.
- **192.168.50.101**—The ACL, in this case an IP address. The ACL can be one of the following:
  - Name—ACL created by name, as described in the "Access Control Lists" section on page 28-8.
  - IP address, as in the example.
  - Network address, including mask; for example, **192.168.50.0/24**.
  - TSIG key—Transaction signature key, in the form **key=***key*, (as described in the "Transaction Security" section on page 28-9.
  - One of the reserved words:

    any—Any ACL

    none—No ACL

    localhost—Any local host addresses

    localnets—Any local network address

  You can negate the ACL value by preceding it with an exclamation point (**!**).
- **name**—Keyword, or type of check to perform on the RR, which can be one of the following:
  - **name**—Name of the RR, requiring a name value.
  - **subdomain**—Name of the RR or the subdomain with any of its RRs, requiring a name or subdomain value.
  - **wildcard**—Name of the RR, using a wildcard value (see Table 28-2 on page 28-13).
- **host1**—Value based on the keyword, in this case the RR named host1. This can also be a subdomain name or, if the **wildcard** keyword is used, can contain wildcards (see Table 28-2 on page 28-13).
- **A,TXT**—RR types, each separated by a comma. This can be a list of any of the RR types described in Appendix A, "Resource Records." You can negate each record type value by preceding it with an exclamation point (**!**).
- Note that if this or any assigned rule is not satisfied, the default is to deny all RR updates.

Tacked onto the end of the rule, outside the quotes, is an index number, in the example, **0**. The index numbers start at 0. If there are multiple rules for an update policy, the index serves to add the rule in a specific order, such that lower numbered indexes have priority in the list. If a rule does not include an index, it is placed at the end of the list. Thus, a rule always has an index, whether or not it is explicitly defined. You also specify the index number in case you need to remove the rule.

To replace a rule, use **update-policy** *name* **delete**, then recreate the update policy. To edit a rule, use **update-policy** *name* **rules remove** *index*, where *index* is the explicitly defined or system-defined index number (remembering that the index numbering starts at 0), then recreate the rule. To remove the second rule in the previous example, enter:

```
nrcmd> update-policy policy1 rules remove 1
```

Applying Update Policies to Zones

After creating an update policy, you can apply it to a zone (forward and reverse) or zone template.

Local Advanced Web UI

Step 1 From the **DNS** menu, choose **Forward Zones** to open the List/Add Zones page.

Step 2 Click the name of the zone to open the Edit Zone page.

Tip You can also perform this function for zone templates on the Edit Zone Template page, and primary reverse zones on the Edit Primary Reverse Zone page (see Chapter 15, "Managing Zones.").

Step 3 Enter the name or (comma-separated) names of one or more of the existing named update policies in the *update-policy-list* attribute field.

Note The server processes the *update-acl* before it processes the *update-policy-list*.

Step 4 Click **Modify Zone**.

CLI Commands

Use **zone** *name* **set update-policy-list**, equating the *update-policy-list* attribute with a quoted list of comma-separated update policies, as defined in the "Creating and Editing Update Policies" section on page 28-12. For example:

```
nrcmd> zone example.com set update-policy-list="policy1,policy2"
```

Confirming Dynamic Records

The Cisco Network Registrar DHCP server stores all pending DNS update data on disk. If the DHCP server cannot communicate with a DNS server, it periodically tests for re-established communication and submits all pending updates. This test typically occurs every 40 seconds.

Local Advanced Web UI

Click **DNS**, then **Forward Zones**. Click the View icon ( ) in the RRs column to open the List/Add DNS Server RRs for Zone page.

CLI Commands

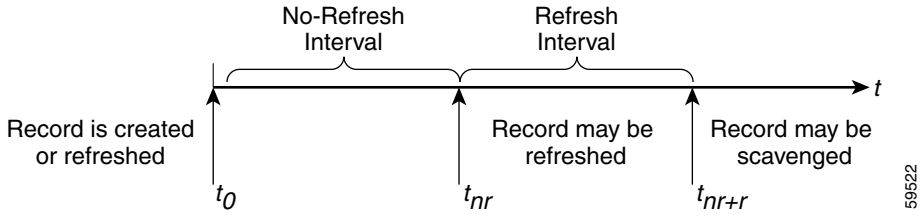
Use **zone** *name* **listRR dns**.

Scavenging Dynamic Records

Microsoft Windows DNS clients that get DHCP leases can update (refresh) their Address (A) records directly with the DNS server. Because many of these clients are mobile laptops that are not permanently connected, some A records may become obsolete over time. The Windows DNS server scavenges and purges these primary zone records periodically. Cisco Network Registrar provides a similar feature that you can use to periodically purge stale records.

Scavenging is normally disabled by default, but you should enable it for zones that exclusively contain Windows clients. Zones are configured with *no-refresh* and *refresh* intervals. A record expires once it ages past its initial creation date plus these two intervals. Figure 28-1 shows the intervals in the scavenging time line.

Figure 28-1 Address Record Scavenging Time Line Intervals



The Cisco Network Registrar process is:

1. When the client updates the DNS server with a new A record, this record gets a timestamp, or if the client refreshes its A record, this may update the timestamp (“Record is created or refreshed”).
2. During a no-refresh interval (a default value of seven days), if the client keeps sending the same record without an address change, this does not update the record timestamp.
3. Once the record ages past the no-refresh interval, it enters the refresh interval (also a default value of seven days), during which time DNS updates refresh the timestamp and put the record back into the no-refresh interval.
4. A record that ages past the refresh interval is available for scavenging when it reaches the scavenge interval.

Note Only unprotected RRs are scavenged. To keep RRs from being scavenged, set them to protected. However, top-of-zone (@) RRs, even if unprotected, are not scavenged.

The following zone attributes affect scavenging:

- ***scvg-interval***—Period during which the DNS server checks for stale records in a zone. The value can range from one hour to 365 days. You can also set this for the server (the default value is one week), although the zone setting overrides it.
- ***scvg-no-refresh-interval***—Interval during which actions, such as dynamic or prerequisite-only DNS updates, do not update the record timestamp. The value can range from one hour to 365 days. The zone setting overrides the server setting (the default value is one week).
- ***scvg-refresh-interval***—Interval during which DNS updates increment the 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. The zone setting overrides the server setting (the default value is one week).
- ***scvg-ignore-restart-interval***—Ensures that the server does not reset the scavenging time with every server restart. Within this interval, Cisco Network Registrar ignores the duration between a server down instance and a restart, which is usually fairly short.

  The value can range from two hours to one day. With any value longer than that set, Cisco Network Registrar recalculates the scavenging period to allow for record updates that cannot take place while the server is stopped. The zone setting overrides the server setting (the default value is 2 hours.

Enable scavenging only for zones where a Cisco Network Registrar DNS server receives updates exclusively from Windows clients (or those known to do automatic periodic DNS updates). Set the attributes listed above. The Cisco Network Registrar scavenging manager starts at server startup. It reports records purged through scavenging to the changeset database. Cisco Network Registrar also notifies secondary zones by way of zone transfers of any records scavenged from the primary zone. In cases where you create a zone that has scavenging disabled (the records do not have a timestamp) and then subsequently enable it, Cisco Network Registrar uses a proxy timestamp as a default timestamp for each record.

You can monitor scavenging activity using one or more of the log settings scavenge, scavenge-details, ddns-refreshes, and ddns-refreshes-details.

Local Advanced Web UI

On the Manage DNS Server page, click the Run icon ( ) in the Commands column to open the DNS Server Commands page (see Figure 7-1 on page 7-2). On this page, click the Run icon next to Scavenge all zones.

To scavenge a particular forward or reverse zone only, go to the Zone Commands for Zone page, which is available by clicking the Run icon ( ) on the List/Add Zones page or List/Add Reverse Zones page. Click the Run icon again next to Scavenge zone on the Zone Commands for Zone page. To find out the next time scavenging is scheduled for the zone, click the Run icon next to Get scavenge start time.

CLI Commands

Use **dns scavenge** for all zones that have scavenging enabled, or **zone** *name* **scavenge** for a specific zone that has it enabled. Use the **getScavengeStartTime** action on a zone to find out the next time scavenging is scheduled to start.

Troubleshooting DNS Update

You can use a standard DNS tool such as **dig** and **nslookup** to query the server for RRs. The tool can be valuable in determining whether dynamically generated RRs are present. For example:

```
$ nslookup
default Server: server2.example.com
Address: 192.168.1.2
> leasehost1.example.com
Server: server2.example.com
Address: 192.168.1.100
> set type=ptr
> 192.168.1.100
Server: server2.example.com
Address: 192.168.1.100

100.40.168.192.in-addr.arpa name = leasehost1.example.com
40.168,192.in-addr.arpa nameserver = server2.example.com
```

You can monitor DNS updates on the DNS server by setting the *log-settings* attribute to *ddns*, or show even more details by setting it to *ddns-details*.

Configuring DNS Update for Windows Clients

The Windows operating system rely heavily on DNS and, to a lesser extent, DHCP. This reliance requires careful preparation on the part of network administrators prior to wide-scale Windows deployments. Windows clients can add entries for themselves into DNS by directly updating forward zones with their address (A) record. They cannot update reverse zones with their pointer (PTR) records.

See Also

Client DNS Updates
Dual Zone Updates for Windows Clients, page 28-20
DNS Update Settings in Windows Clients, page 28-21
Windows Client Settings in DHCP Servers, page 28-21
SRV Records and DNS Updates, page 28-22
Issues Related to Windows Environments, page 28-23
Frequently Asked Questions About Windows Integration, page 28-27

Client DNS Updates

It is not recommended that clients be allowed to update DNS directly.

For a Windows client to send address record updates to the DNS server, two conditions must apply:

- The Windows client must have the **Register this connection's addresses in DNS** box checked on the **DNS** tab of its TCP/IP control panel settings.
- The DHCP policy must enable direct updating (Cisco Network Registrar policies do so by default).

The Windows client notifies the DHCP server of its intention to update the A record to the DNS server by sending the *client-fqdn* DHCP option (81) in a DHCPREQUEST packet. By indicating the fully qualified domain name (FQDN), the option states unambiguously the client location in the domain namespace. Along with the FQDN itself, the client or server can send one of these possible flags in the *client-fqdn* option:

- **0**—Client should register its A record directly with the DNS server, and the DHCP server registers the PTR record (done through the policy *allow-client-a-record-update* attribute being enabled).
- **1**—Client wants the DHCP server to register its A and PTR records with the DNS server.
- **3**—DHCP server registers the A and PTR records with the DNS server regardless of the client request (done through the policy *allow-client-a-record-update* attribute being disabled, which is the default value). Only the DHCP server can set this flag.

The DHCP server returns its own *client-fqdn* response to the client in a DHCPACK based on whether DNS update is enabled. However, if the 0 flag is set (the *allow-client-a-record-update* attribute is enabled for the policy), enabling or disabling DNS update is irrelevant, because the client can still send its updates to DNS servers. See Table 28-3 on page 28-19 for the actions taken based on how various properties are set.

Table 28-3 Windows Client DNS Update Options

DHCP Client Action	DNS Update	DHCP Server Action
Checks **Register this connection's addresses in DNS** and sends *client-fqdn*; DHCP server enables *allow-client-a-record-update*	Enabled or disabled	Responds with *client-fqdn* that it allows the client to update its A records (sets flag 0), but the DHCP server still updates the PTR records.
Checks **Register**... and sends *client-fqdn*; DHCP disables *allow-client-a-record-update*	Enabled	Responds with *client-fqdn* that it does not allow the client to update the DNS server directly (sets flag 3), and updates the A and PTR records.
	Disabled	Does not respond with *client-fqdn* and does not update the DNS server.
Unchecks **Register**... and sends *client-fqdn*	Enabled	Responds with *client-fqdn* that it is updating the A and PTR records.
	Disabled	Does not respond with *client-fqdn* and does not update the DNS server.
Does not send *client-fqdn*	Enabled	Does not respond with *client-fqdn*, but updates the A and PTR records.
	Disabled	Does not respond with *client-fqdn* and does not update the DNS server.

A Windows DHCP server can set the *client-fqdn* option to ignore the client request. To enable this behavior in Cisco Network Registrar, create a policy for Windows clients and disable the *allow-client-a-record-update* attribute for this policy.

The following attributes are enabled by default in Cisco Network Registrar:

- **Server *use-client-fqdn***—The server uses the *client-fqdn* value on incoming packets and does not examine the *host-name*. The DHCP server ignores all characters after the first dot in the domain name value, because it determines the domain from the defined scope for that client. Disable *use-client-fqdn* only if you do not want the server to determine hostnames from *client-fqdn*, possibly because the client is sending unexpected characters.

- **Server *use-client-fqdn-first***—The server examines *client-fqdn* on incoming packets from the client before examining the *host-name* option (12). If *client-fqdn* contains a hostname, the server uses it. If the server does not find the option, it uses the *host-name* value. If *use-client-fqdn-first* is disabled, the server prefers the *host-name* value over *client-fqdn*.
- **Server *use-client-fqdn-if-asked***—The server returns the *client-fqdn* value in the outgoing packets if the client requests it. For example, the client might want to know the status of DNS activity, and hence request that the DHCP server should present the *client-fqdn* value.
- **Policy *allow-client-a-record-update***—The client can update its A record directly with the DNS server, as long as the client sets the *client-fqdn* flag to 0 (requesting direct updating). Otherwise, the server updates the A record based on other configuration properties.

The hostnames returned to client requests vary depending on these settings (see Table 28-4 on page 28-20).

Table 28-4 Hostnames Returned Based on Client Request Parameters

Client Request	With Server/Policy Settings	Resulting Hostname
Includes *host-name* (option 12)	*use-host-name*=true *use-client-fqdn*=false (or *use-client-fqdn-first*=false) *trim-host-name*=true	*host-name* trimmed at first dot. Example: *host-name* host1.bob is returned host1.
	(same except:) *trim-host-name*=false	*host-name*. Example: *host-name* host1.bob is returned host1.bob.
Includes *client-fqdn* (option 81)	*use-client-fqdn*=true *use-host-name*=false (or *use-client-fqdn-first*=true)	*client-fqdn* trimmed at first dot. Example: *client-fqdn* host1.bob is returned host1.
Omits *host-name* (option 12) and *client-fqdn* (option 81)	Or: *use-host-name*=false *use-client-fqdn*=false	Set by client/policy hierarchy.
	(same as the previous except:) hostname is undefined in the client/policy hierarchy, with *synthesize-name*=true	Synthesized following the synthesizing rule, which is to append the hyphenated IP address of the host after the specified *synthetic-name-stem*.
	(same as the previous except:) *synthesize-name*=false	Undefined.

Dual Zone Updates for Windows Clients

Windows DHCP clients might be part of a DHCP deployment where they have A records in two DNS zones. In this case, the DHCP server returns the *client-fqdn* so that the client can request a dual zone update. To enable a dual zone update, enable the policy attribute *allow-dual-zone-dns-update*.

The DHCP client sends the 0 flag in *client-fqdn* and the DHCP server returns the 0 flag so that the client can update the DNS server with the A record in its main zone. However, the DHCP server also directly sends an A record update based on the client secondary zone in the behalf of the client. If both *allow-client-a-record-update* and the *allow-dual-zone-dns-update* are enabled, allowing the dual zone update takes precedence so that the server can update the secondary zone A record.

DNS Update Settings in Windows Clients

The Windows client can set advanced properties to enable sending the *client-fqdn* option.

Step 1 On the Windows client, go to the Control Panel and open the TCP/IP Settings dialog box.

Step 2 Click the **Advanced** tab.

Step 3 Click the **DNS** tab.

Step 4 To have the client send the *client-fqdn* option in its request, leave the **Register this connection's addresses in DNS** box checked. This indicates that the client wants to do the A record update.

Windows Client Settings in DHCP Servers

You can apply a relevant policy to a scope that includes the Windows clients, and enable DNS updates for the scope.

Step 1 Create a policy for the scope that includes the Windows clients. For example:

a. Create a policywin2k.

b. Create a win2k scope with the subnet 192.168.1.0/24 and policywin2k as the policy. Add an address range of 192.168.1.10 through 192.168.1.100.

Step 2 Set the scope attribute *dynamic-dns* to update-all, update-fwd-only, or update-rev-only.

Step 3 Set the zone name, server address (for A records), reverse zone name, and reverse server address (for PTR records), as described in the "Creating DNS Update Configurations" section on page 28-5.

Step 4 If you want the client to update its A records at the DNS server, enable the policy attribute *allow-client-a-record-update* (this is the preset value). There are a few caveats to this:

- If *allow-client-a-record-update* is enabled and the client sends the *client-fqdn* with the update bit enabled, the *host-name* and *client-fqdn* returned to the client match the client *client-fqdn*. (However, if the *override-client-fqdn* is also enabled on the server, the hostname and FQDN returned to the client are generated by the configured hostname and policy domain name.)
- If, instead, the client does not send the *client-fqdn* with the update bit enabled, the server does the A record update, and the *host-name* and *client-fqdn* (if requested) returned to the client match the name used for the DNS update.
- If *allow-client-a-record-update* is disabled, the server does the A record updates, and the *host-name* and *client-fqdn* (with the update bit disabled) values returned to the client match the name used for the DNS update.
- If *allow-dual-zone-dns-update* is enabled, the DHCP server always does the A record updates. (See the "Dual Zone Updates for Windows Clients" section on page 28-20.)
- If *use-dns-update-prereqs* is enabled (the preset value) for the DHCP server or DNS update configuration and *update-dns-first* is disabled (the preset value) for the update configuration, the hostname and *client-fqdn* returned to the client are not guaranteed to match the DNS update, because of delayed name disambiguation. However, the lease data will be updated with the new names.

  According to RFC 2136, update prerequisites determine the action the primary master DNS server takes based on whether an RR set or name record should or should not exist. Disable *use-dns-update-prereqs* only under rare circumstances.

Step 5 Reload the DHCP server.

SRV Records and DNS Updates

Windows relies heavily on the DNS protocol for advertising services to the network. Table 28-5 on page 28-22 describes how Windows handles service location (SRV) DNS RRs and DNS updates.

You can configure the Cisco Network Registrar DNS server so that Windows domain controllers can dynamically register their services in DNS and, thereby, advertise themselves to the network. Because this process occurs through RFC-compliant DNS updates, you do not need to do anything out of the ordinary in Cisco Network Registrar.

Table 28-5 Windows SRV Records and DNS Updates

Feature	Description

SRV records	Windows domain controllers use the SRV RR to advertise services to the network. This RR is defined in the RFC 2782, "A DNS RR for specifying the location of services (DNS SRV)." The RFC defines the format of the SRV record (DNS type code 33) as:
	_service._protocol.name ttl class SRV *priority weight port target*
	There should always be an A record associated with target of the SRV record, so that the client can resolve the service back to a host. In the Windows implementation of SRV records, the records might look like this:
	`myserver.example.com A 10.100.200.11`
	`_ldap._tcp.example.com SRV 0 0 389 myserver.example.com`
	`_kdc._tcp.example.com SRV 0 0 88 myserver.example.com`
	`_ldap._tcp.dc_msdcs.example.com SRV 0 0 88 myserver.example.com`
	An underscore always precedes the service and protocol names. In the example, _kdc is the Key Distribution Center. The priority and weight help you choose between target servers providing the same service (the weight differentiating those with equal priorities). With zero priority and weight, the listed order determines the priority. Windows domain controllers automatically place these SRV records in DNS.

How SRV records are used	When a Windows client boots up, it tries to initiate the network login process to authenticate against its Windows domain controller. The client must first discover where the domain controller is, and they do so using the dynamically generated SRV records. Before launching the net-login process, the client queries DNS with a service name; for example, _ldap._tcp.dc_msdcs.example.com. The DNS server SRV record target, for example, is my-domain-controller.example.com. The Windows client then queries DNS with the hostname my-domain-controller.example.com. DNS returns the host address and the client uses this address to find the domain controller. The net-login process fails without these SRV records.
DNS updates	When a Windows server is configured as a domain controller, you statically configure the name of the domain it manages through the Active Directory management console. This Windows domain should have a corresponding DNS zone associated with it. The domain controller should also have a series of DNS resolvers configured in its TCP/IP properties control panel.

Table 28-5 Windows SRV Records and DNS Updates (continued)

Feature	Description
	When the Windows domain controller boots up, it performs these steps to register itself in DNS and advertise its services to the network: 1. Queries DNS asking for the start of authority (SOA) record for the DNS domain that mostly closely encapsulates its Windows domain. 2. Identifies the primary DNS server for the DNS zone (from the SOA record) that mostly closely encapsulates its Windows domain name. 3. Creates a series of SRV records in this zone using the RFC 2136 DNS Update protocol.

Server boot process log file examples	Under normal operating conditions, the Cisco Network Registrar primary DNS server writes these log entries when a Windows domain controller boots up and creates its SRV records:
	`data time name/dns/1 Activity Protocol 0 Added type 33 record to name "_ldap._tcp.w2k.example.com", zone "w2k.example.com"`
	`data time name/dns/1 Activity Protocol 0 Update of zone "w2k.example.com" from address [10.100.200.2] succeeded.`
	This log shows only one DNS update for a single SRV record. A Windows domain controller typically registers 17 of these SRV records when it boots up.

To configure Cisco Network Registrar to accept these dynamic SRV record updates:

Step 1 Determine the IP addresses of the devices in the network that need to advertise services through DNS.

Step 2 If they do not exist, create the appropriate forward and reverse zones for the Windows domains.

Step 3 Enable DNS updates for the forward and reverse zones.

Step 4 Set up a DNS update policy to define the IP addresses of the hosts to which you want to restrict accepting DNS updates (see the "Configuring DNS Update Policies" section on page 28-12). These are usually the DHCP servers and any Windows domain controllers. (The Windows domain controllers should have static IP addresses.)

If it is impractical or impossible to enter the list of all the IP addresses from which a DNS server must accept updates, you can configure Cisco Network Registrar to accept updates from a range of addresses, although Cisco does not recommend this configuration.

Step 5 Reload the DNS and DHCP servers.

Issues Related to Windows Environments

Table 28-6 describes the issues concerning interoperability between Windows and Cisco Network Registrar, The information in this table is intended to inform you of possible problems before you encounter them in the field. For some frequently asked questions about Windows interoperability, see the "Frequently Asked Questions About Windows Integration" section on page 28-27.

Table 28-6 Issues Concerning Windows and Cisco Network Registrar Interoperability

Issue	Description

Invisible dynamically created RRs	Cisco Network Registrar, when properly configured, accepts DNS updates from both DHCP and Windows servers. You can use the CLI to access the dynamic portion of the DNS zone for viewing and deleting records. Enter this command to view all DNS RRs in a given zone:
	`nrcmd> zone myzone listRR dynamic myfile`
	This redirects the output to the myfile file (see Example 28-1 on page 28-26).
	You can delete dynamically generated records by entering this command:
	`nrcmd> zone myzone removeDynRR myname [type]`
	You can also use **nslookup** to verify their existence, and you can use version 5.*x* (shipped with Windows) to view dynamic SRV records. In this version, use **set type=SRV** to enable viewing SRV records.

Domain controller registration	A Windows domain controller has to register itself in DNS using DNS updates. The DNS RFCs dictate that only a primary DNS server for a particular zone can accept edits to the zone data. Hence, the Windows domain controller has to discover which DNS server is the primary for the zone that includes its Windows domain name. The domain controller discovers this by querying the first DNS server in its resolver list (configured in the TCP/IP properties control panel). The initial query is for the SOA record of the zone that includes the Windows domain of the domain controller. The SOA record includes the name of the primary server for the zone. If no zone exists for the domain name, the domain controller keeps removing the left-most label of the domain name and sends queries until it finds an SOA record with a primary server included in that domain. Once the domain controller has the name of the primary DNS server for its domain, it sends it DNS updates to create the necessary SRV records. Ensure that the name of the zone primary DNS server is in its SOA record.
Failure of A record DNS updates	When a Windows domain controller tries to advertise itself to the network, it sends several DNS update requests to the DNS server of record for its domain. Most of these update requests are for SRV records. However, the domain controller also requests an update for a single A record of the same name as the Windows domain. If the Cisco Network Registrar DNS server is also authoritative for a zone identical to this Windows domain, it rejects registering its A record, because the DNS A record update conflicts with the static SOA and NS records. This is to prevent possible security infractions, such as a dynamic host registering itself and spoofing Web traffic to a site. For example, the domain controller might control the w2k.example.com Windows zone. If a zone with the same name exists on the Cisco Network Registrar DNS server, these RRs could be part of that zone. (Example follows.)

Table 28-6 Issues Concerning Windows and Cisco Network Registrar Interoperability (continued)

Issue	Description
	w2k.example.com. 43200 SOA nameserver.example.com. hostmaster.example.com. (98011312 ;serial 3600 ;refresh 3600 ;retry 3600000 ;expire 43200) ;minim w2k.example.com.86400 NS nameserver.example.com The domain controller would try to add an additional record; for example: w2k.example.com. 86400 A 192.168.2.1 Cisco Network Registrar does not allow a DNS update to conflict with any statically configured name in the zone, even if the record type associated with that name is different. In the above example, an attempt to add an A record associated with the name w2k.example.com collides with the SOA and NS records. When the domain controller boots up, a DNS log file entry such as this appears: 08/10/2000 16:35:33 name/dns/1 Info Protocol 0 Error - REFUSED - Update of static name "w2k.example.com", from address [10.100.200.2] This is how Cisco Network Registrar responds to DNS updates of static DNS data. Additionally, you can ignore this DNS update failure. Windows clients do not use this A record. Allocation of domain controllers happens through SRV records. Microsoft added the A record to accommodate legacy NT clients that do not support SRV records. Note that failing to register the controller A record slows down the domain controller bootup process, affecting the overall login of worker clients. As mentioned earlier, the workaround is to define the Windows domain as a subdomain of the authoritative zone, or enable the DNS server *simulate-zone-top-dynupdate* attribute. If this is not possible, contact the Cisco Technical Assistance Center for help.
Windows RC1 DHCP clients	Microsoft released Windows build 2072 (known as RC1) with a broken DHCP client. This client sends a misformed packet that Cisco Network Registrar cannot parse. Cisco Network Registrar drops the packet and cannot serve the client, logging this error: 08/10/2000 14:56:23 name/dhcp/1 Activity Protocol 0 10.0.0.15 Lease offered to Host:'My-Computer' CID: 01:00:a0:24:1a:b0:d8 packet'R15' until True, 10 Aug 2000 14:58:23 -0400. 301 ms. 08/10/2000 14:56:23 name/dhcp/1 Warning Protocol 0 Unable to find necessary Client information in packet from MAC address:'1,6,00:d0:ba:d3:bd:3b'. Packet dropped! Cisco Network Registrar includes error checking specifically designed to deal with errors such as this improperly built FQDN option. However, if you encounter this problem, install the Microsoft patch to the RC1 client on the DHCP client. You must obtain this patch from Microsoft.

Table 28-6 Issues Concerning Windows and Cisco Network Registrar Interoperability (continued)

Issue	Description
Windows plug-and-play network interface card (NIC) configuration	If configured to use DHCP, a Windows system tries to obtain a DHCP lease on startup. If no DHCP server is available, Windows may automatically configure the computer interface with a plug-and-play IP address. This address is not one that the network administrator or DHCP server configured or selected. These plug-and-play addresses are in the range 169.254.0.0/16. If you see devices in this address range on a network, it means that Windows autoconfigured the interfaces because it could not obtain a lease from a DHCP server. This can cause significant network and troubleshooting problems. The Windows system no longer informs the user that the DHCP client could not obtain a lease. Everything appears to function normally, but the client cannot route packets off its local subnet. Additionally, you may see the DHCP client trying to operate on the network with an address from the 169.254.0.0/16 network. This may lead you to think that the Cisco Network Registrar DHCP server is broken and handing out the wrong addresses. If this problem occurs, perform these steps: 1. Ensure that the DHCP client has an active network port and a properly configured NIC. 2. Ensure that the network between the client and the DHCP server(s) are properly configured. Ensure that all router interfaces are configured with the correct IPHelper address. 3. Reboot the DHCP client. 4. If necessary, look at the DHCP log file. If the DHCP client can successfully route packets to the server, this logs a DHCPDISCOVER, even if Cisco Network Registrar does not answer the packet. If the network is correctly configured, and if the DHCP client is not broken, Cisco Network Registrar should receive the packet and log it. If there is no log entry for a packet receipt, there is a problem somewhere else in the network.
Scavenging Windows client address records	Windows clients do not clean up after themselves, potentially causing their dynamic record registration to remain indefinitely. This leaves stale address records on the DNS server. To ensure that these stale records are periodically removed, you must enable scavenging for the zone (see the "Scavenging Dynamic Records" section on page 28-16).

Example 28-1 Output Showing Invisible Dynamically Created RRs

```
Dynamic Resource Records
_ldap._tcp.test-lab._sites 600 IN SRV 0 100 389 CNR-MKT-1.w2k.example.com.
_ldap._tcp.test-lab._sites.gc._msdcs 600 IN SRV 0 100 3268 CNR-MKT-1.w2k.example.com.
_kerberos._tcp.test-lab._sites.dc._msdcs 600 IN SRV 0 100 88 CNR-MKT-1.w2k.example.com.
_ldap._tcp.test-lab._sites.dc._msdcs 600 IN SRV 0 100 389 CNR-MKT-1.w2k.example.com.
_ldap._tcp 600 IN SRV 0 100 389 CNR-MKT-1.w2k.example.com.
_kerberos._tcp.test-lab._sites 600 IN SRV 0 100 88 CNR-MKT-1.w2k.example.com.
_ldap._tcp.pdc._msdcs 600 IN SRV 0 100 389 CNR-MKT-1.w2k.example.com.
_ldap._tcp.gc._msdcs 600 IN SRV 0 100 3268 CNR-MKT-1.w2k.example.com.
_ldap._tcp.1ca176bc-86bf-46f1-8a0f-235ab891bcd2.domains._msdcs 600 IN SRV 0 100 389
    CNR-MKT-1.w2k.example.com.
e5b0e667-27c8-44f7-bd76-6b8385c74bd7._msdcs 600 IN CNAME CNR-MKT-1.w2k.example.com.
```

```
_kerberos._tcp.dc._msdcs 600 IN SRV 0 100 88 CNR-MKT-1.w2k.example.com.
_ldap._tcp.dc._msdcs 600 IN SRV 0 100 389 CNR-MKT-1.w2k.example.com.
_kerberos._tcp  600 IN SRV 0 100 88 CNR-MKT-1.w2k.example.com.
_gc._tcp 600 IN SRV 0 100 3268 CNR-MKT-1.w2k.example.com.
_kerberos._udp 600 IN SRV 0 100 88 CNR-MKT-1.w2k.example.com.
_kpasswd._tcp 600 IN SRV 0 100 464 CNR-MKT-1.w2k.example.com.
_kpasswd._udp 600 IN SRV 0 100 464 CNR-MKT-1.w2k.example.com.
gc._msdcs 600 IN A 10.100.200.2
_gc._tcp.test-lab._sites 600 IN SRV 0 100 3268 CNR-MKT-1.w2k.example.com.
```

Frequently Asked Questions About Windows Integration

These questions are frequently asked about integrating Cisco Network Registrar DNS services with Windows:

Q. What happens if both Windows clients and the DHCP server are allowed to update the same zone? Can this create the potential for stale DNS records being left in a zone? If so, what can be done about it?

A. The recommendation is not to allow Windows clients to update their zones. Instead, the DHCP server should manage all the client dynamic RR records. When configured to perform DNS updates, the DHCP server accurately manages all the RRs associated with the clients that it served leases to. In contrast, Windows client machines blindly send a daily DNS update to the server, and when removed from the network, leave a stale DNS entry behind.

Any zone being updated by DNS update clients should have DNS scavenging enabled to shorten the longevity of stale RRs left by transient Windows clients. If the DHCP server and Windows clients are both updating the same zone, three things are required in Cisco Network Registrar:

a. Enable scavenging for the zone.

b. Configure the DHCP server to refresh its DNS update entries as each client renews its lease. By default, Cisco Network Registrar does not update the DNS record again between its creation and its final deletion. A DNS update record that Cisco Network Registrar creates lives from the start of the lease until the lease expires. You can change this behavior using a DHCP server (or DNS update configuration) attribute, *force-dns-updates*. For example:

```
nrcmd> dhcp enable force-dns-updates
100 Ok
force-dns-updates=true
```

c. If scavenging is enabled on a particular zone, then the lease time associated with clients that the DHCP server updates that zone on behalf of must be less than the sum of the *no-refresh-interval* and *refresh-interval* scavenging settings. Both of these settings default to seven days. You can set the lease time to 14 days or less if you do not change these default values.

Q. What needs to be done to integrate a Windows domain with a pre-existing DNS domain naming structure if it was decided not to have overlapping DNS and Windows domains? For example, if there is a pre-existing DNS domain called example.com and a Windows domain is created that is called w2k.example.com, what needs to be done to integrate the Windows domain with the DNS domain?

A. In the example, a tree in the Windows domain forest would have a root of w2k.example.com. There would be a DNS domain named example.com. This DNS domain would be represented by a zone named example.com. There may be additional DNS subdomains represented in this zone, but no subdomains are ever delegated out of this zone into their own zones. All the subdomains will always reside in the example.com. zone.

Q. *In this case, how are DNS updates from the domain controllers dealt with?*

A. To deal with the SRV record updates from the Windows domain controllers, limit DNS updates to the example.com. zone to the domain controllers by IP address only. (Later, you will also add the IP address of the DHCP server to the list.) Enable scavenging on the zone. The controllers will update SRV and A records for the w2k.example.com subdomain in the example.com zone. There is no special configuration required to deal with the A record update from each domain controller, because an A record for w2k.example.com does not conflict with the SOA, NS, or any other static record in the example.com zone.

The example.com zone then might include these records:

```
example.com. 43200 SOA ns.example.com. hostmaster.example.com. (
    98011312 ;serial
    3600 ;refresh
    3600 ;retry
    3600000 ;expire
    43200 ) ;minimum
example.com.86400 NS ns.example.com
ns.example.com. 86400 A 10.0.0.10
_ldap._tcp.w2k.example.com. IN SRV 0 0 389 dc1.w2k.example.com
w2k.example.com 86400 A 10.0.0.25
...
```

Q. *In this case, how are zone updates from individual Windows client machines dealt with?*

A. In this scenario, the clients could potentially try to update the example.com. zone with updates to the w2k.example.com domain. The way to avoid this is to close down the zone to updates except from trusted sources. For Cisco Network Registrar 7.2, you can use transaction signatures (TSIG) between the DHCP server and the primary DNS server for the example.com zone.

Configure the DHCP server to do DNS updates to the example.com zone and the appropriate reverse zone for each client, and use option 81 to prevent the clients from doing DNS updates themselves.

Q. *Has security been addressed in this case?*

A. By configuring the forward and reverse zone to accept only updates from trusted IP addresses, you close the zone to updates from any other device on the network. Security by IP is not the most ideal solution, as it would not prevent a malicious attack from a spoofed IP address source. You can secure updates from the DHCP server by configuring TSIG between the DHCP server and the DNS server.

Q. *Is scavenging required in this case?*

A. No. Updates are only accepted from the domain controllers and the DHCP server. The DHCP server accurately maintains the life cycle of the records that they add and do not require scavenging. You can manage the domain controller dynamic entries manually by using the Cisco Network Registrar single-record dynamic RR removal feature.

Q. What needs to be done to integrate a Windows domain that shares its namespace with a DNS domain? For example, if there is a pre-existing DNS zone called example.com and a Windows Active Directory domain called example.com needs to be deployed, how can it be done?

A. In this example, a tree in the Windows domain forest would have a root of example.com. There is a pre-existing domain that is also named example.com that is represented by a zone named example.com.

Q. *In this case, how are DNS updates from individual Windows client machines dealt with?*

A. To deal with the SRV record updates, create subzones for:

```
_tcp.example.com.
_sites.example.com.
_msdcs.example.com.
_msdcs.example.com.
_udp.example.com.
```

Limit DNS updates to those zones to the domain controllers by IP address only. Enable scavenging on these zones.

To deal with the A record update from each domain controller, enable a DNS server attribute, *simulate-zone-top-dynupdate*.

```
nrcmd> dns enable simulate-zone-top-dynupdate
```

It is not required, but if desired, manually add an A record for the domain controllers to the example.com zone.

Q. *In this case, how are zone updates from individual Windows client machines dealt with?*

A. In this scenario, the clients could potentially try to update the example.com zone. The way to avoid this is to close down the zone to updates except from trusted sources. For Cisco Network Registrar 7.2, you can use transaction signatures (TSIG) between the DHCP server and the primary DNS server for the example.com zone.

Configure the DHCP server to do DNS updates to the example.com zone and the appropriate reverse zone for each client, and use option 81 to prevent the clients from doing DNS updates themselves.

Q. *Has security been addressed in this case?*

A. By configuring the forward and reverse zone to accept only updates from trusted IP addresses, you close the zone to updates from other devices on the network. Security by IP is not the most ideal solution, as it would not prevent a malicious attack from a spoofed source. Updates from the DHCP server are more secure when TSIG is configured between the DHCP server and the DNS server.

Q. *Has scavenging been addressed in this case?*

A. Yes. The subzones _tcp.example.com, _sites.example.com, _msdcs.example.com, _msdcs.example.com, and _udp.example.com zones accept updates only from the domain controllers, and scavenging was turned on for these zones. The example.com zone accepts DNS updates only from the DHCP server.