Technical Guide

Domain Name System (DNS) for AlliedWare Plus™ AR-Series Firewalls

Feature Overview and Configuration Guide

Introduction

The **Domain Name System** allows you to access remote systems by entering human-readable device host names rather than IP addresses. DNS works by creating a mapping between a domain name, such as ‘www.alliedtelesis.com’, and its IP address. These mappings are held on DNS servers. DNS translates meaningful domain names into IP addresses for networking equipment to locate and address these devices. The benefits of DNS are that domain names:

- can map to a new IP address if the host’s IP address changes
- are easier to remember than an IP address
- allow organizations to use a domain name hierarchy that is independent of any IP address assignment

AlliedWare Plus™ has the ability to resolve IP addresses associated with domain names for internally generated commands (DNS Client) as well as providing the DNS information to connected hosts (via DNS Relay and DHCP Server). The DNS Client is enabled automatically when at least one DNS server is configured on the device. This client allows you to use domain names instead of IP addresses when using commands on your device, like ping, SSH, and copy.

The DNS Relay provides the presence of a local virtual DNS server which can service DNS lookup requests sent to it from local hosts. The DHCP Server can be configured to provide DNS information to DHCP clients during the lease process.

The **Dynamic Domain Name System** (DDNS) is a mechanism which allows a DDNS client to automatically update a DNS entry hosted by a DDNS Provider. When DDNS is configured on an AR-Series Firewall, DNS updates are automatically directed to the configured host name regardless of Dynamic IP address changes. This feature is available on all AR-Series Firewalls from release 5.4.7 onwards.

Products and software version that apply to this guide

This guide applies to all AlliedWare Plus™ AR-Series Firewalls. The following features are supported from the following software versions:

- DDNS Client - 5.4.7-0.1
- DNS Client and DNS Relay - 5.4.4
- DDNS new function to handle redirects and returned parameters.
- The DNS client can now be VRF-aware - 5.5.2-1.1
- Configuration via the Device GUI - version 2.16.0 of the Device GUI

For details about the DNS commands, see the product's Command Reference.

For information about DNS on AlliedWare Plus switches, see Domain Name System (DNS) for AlliedWare Plus Switches.

Contents

Domain Name System (DNS)

Domain name parts

Domain names are made up of a hierarchy of two or more name segments. Each segment is separated by a period. The format of domain names is the same as the host portion of a URL (Uniform Resource Locator). The first segment from the left is unique to the host, with each following segment mapping the host in the domain name hierarchy. The segment on the far right is a top-level domain name shared by many hosts.

Server hierarchy

A network of domain name servers maintains the mappings between domain names and their IP addresses. This network operates in a hierarchy that is similar to the structure of the domain names. When a local DNS server cannot resolve your request it sends the request to a higher level DNS server.

For example, to access the site “alliedtelesis.com”, your PC sends a DNS inquiry to its local DNS server asking for the IP address matching alliedtelesis.com. If this address is already locally cached (following its recent use), the DNS server returns the IP address that matches alliedtelesis.com. If the DNS server does not have this address cached, it forwards the request upwards through the hierarchy of DNS servers until a DNS server can resolve the mapping. This means an often-used domain name is resolved quickly, while an uncommon or nonexistent domain may take longer to resolve or fail.

As well as the hierarchy of domain name servers accessible through the Internet, you can operate your own DNS server to map to private IP addresses within your network.

The DHCP server IP address can be either statically defined, or can be dynamically assigned via DHCPv4 option 6 using the **ip name-server** command and DHCP option 15 using the **ip domain-name** command if the DHCP client is configured.

Setting a preference between static or dynamic DNS servers

From release 5.4.9-0.1 onwards, it is possible to set a preference between using statically configured DNS servers or dynamically learned DNS servers.

The command **ip name-server preferred-order [*dynamic*|*static*]** can be used to choose which DNS server set to use first. Select either the **dynamic** or **static** parameter.

By default, the dynamic learned DNS servers are used first. For example, if you want to change the preference to use static servers first, use the command, **ip name-server preferred-order static**.

DNS client

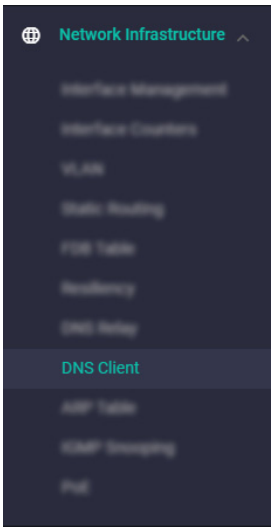
Your AlliedWare Plus device has a DNS Client that is enabled automatically when you configure a name server address on your device. This client allows you to use domain names instead of IP addresses when configuring some features on your device.

DNS client in the Device GUI

From Device GUI version 2.16.0 onwards, you can use the DNS Client page to:

- configure DNS clients,
- add servers to DNS clients, and
- add or remove domains from the domain list.

Use the left-hand menu to navigate to **Network Infrastructure** > **DNS Client**



The DNS Client page contains the Configure button, and DNS Servers and Domain List tables.

Clicking on the **Configuration** button will open the Configure window, where you can configure **Domain Lookup via Relay**. Note that DNS Relay is disabled by default. For more information about DNS Relay, see "Enabling DNS Relay in the Device GUI" on page 8.

The **DNS Servers** table displays the servers that you configure for the client to contact. The DNS Servers table includes a source column, which is the source that it learns the server's IP from.

You can use the **Domain List** table to append domains to the DNS query. This is useful if your network has multiple domains, and you would like to simplify the administration. The domain list is ordered based on the order you enter the domains in. In other words, the first entry you create is queried first.

You may want to use the domain list to filter through specific domains, or top-level domains. For example:

- if you add a domain entry as 'domain.com', and then perform a DNS query for 'site1' it will check **site1.domain.com.**
- if you add a top-level domain entry as '.com', and then you perform a DNS query for 'site1', it will check **site1.com**.

Click **+ New Domain** to add a new domain entry in the list.

DNS client commands

The following section shows you a variety of commands that you can use to set up DNS on your device.

Add DNS Server

To add a DNS server to the list of servers that the device sends DNS queries to, use the command:

```
awplus(config)# ip name-server <ip-addr>
```

The order that you enter the servers in, is the order in which they will be used.

Check list of servers

To check the list of servers that the device sends DNS queries to, use the command:

```
awplus# show ip name-server
```

Add default domain name

To add a default domain name used to append to DNS requests, use the command:

```
awplus(config)# ip domain-name <domain-name>
```

To check the domain name configured with this command, use the show command:

```
awplus# show ip domain-name
```

Matching hostnames to network

To use DNS to match hostnames to your internal network named "example.net", use the command:

```
awplus(config)# ip domain-name example.net
```

Then, if you use the command **ping host2**, your device sends a DNS request for host2.example.net.

Domain List

Alternatively you can create a list of domain names that your device will try in turn by using the command:

```
awplus(config)# ip domain-list <domain-name>
```

For example, to use DNS to match incomplete hostnames to the top level domains ".com", and ".net", use the commands:

```
awplus(config)# ip domain-list .com
```

```
awplus(config)# ip domain-list .net
```

If you then use the command **ping alliedtelesis**, your device sends a DNS request for alliedtelesis.com and if no match was found your device would then try alliedtelesis.net.

Check domain list entries

To check the entries in the domain list, use the command:

```
awplus# show ip domain-list
```

Disable DNS client

To disable the DNS client on your device, use the command:

```
awplus(config)# no ip domain-lookup
```

Check DNS client status

To check the status of the DNS client on your device, and the configured servers and domain names, use the command:

```
awplus# show hosts
```

DNS Relay

Enabling DNS Relay on your device provides the capability for it to act as a local virtual DNS server. You device can then service DNS lookup requests sent to it from local hosts.

When your device receives a DNS query from a client, the device will attempt to match the request with entries in its cache. If the device does not have this address cached, it forwards the request upwards through the hierarchy of DNS servers for resolution.

When acting as a DNS Relay**,** the device will relay (pass on) the requests to an external, or upstream, DNS server. The relaying of DNS queries is useful if a network administrator wishes to easily change to using a different DNS server.

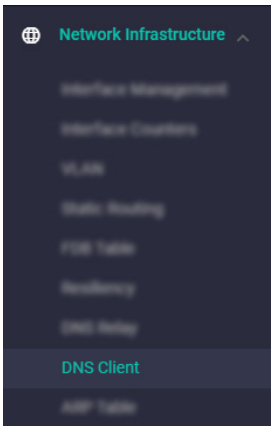
You may wish to configure only the gateway device with the actual DNS address(es), and then configure all the other devices to send their DNS requests to that gateway device. This would mean that, when changing to a different DNS server, you only need to update the DNS address(es) in one place, on the gateway device. This is far more convenient than having to update DNS addresses in all the individual hosts in the network.

DNS Relay requires that IP domain lookup is enabled. To see how to enable IP domain lookup from the CLI, see "DNS Relay commands" on page 9.

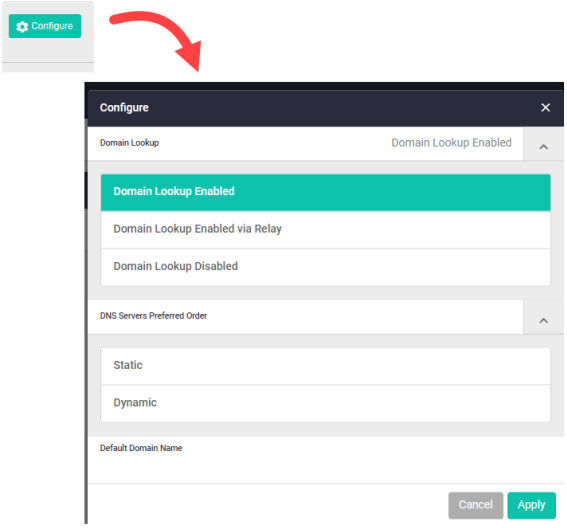
Enabling DNS Relay in the Device GUI

To enable **DNS Relay** from the Device GUI, also known as **Domain Lookup Enabled via Relay,** you must have Device GUI version 2.16.0 or later installed on your device.

Use the left-hand menu to navigate to **Network Infrastructure** > **DNS Client**



Click on the **Configure** button on the DNS Client page to open the Domain Lookup settings:



Toggle the Domain Lookup dropdown, and select **Domain Lookup Enabled via Relay** to enable DNS Relay. You can also select a specific order that you would prefer the relay to occur in.

Enabling IP Name Server for DNS Relay

DNS Relay uses the DNS server list configured by the **ip name-server** command to forward DNS query packets. To enable DNS Relay you need to configure the list of servers that the device sends DNS queries to and then enable DNS forwarding, as shown in the following example for a DNS server with an IPv4 address:

```
awplus# configure terminal
awplus(config)# ip name-server 192.168.1.1
awplus(config)# ip name-server 192.168.1.2
awplus(config)# ip dns forwarding
```

DNS Relay requires that IP domain lookup is enabled.

IP domain lookup is enabled by default, but if it has been disabled, you can re-enable it by using the command:

```
awplus(config)# ip domain-lookup
```

Note: Both IPv4 and IPv6 support DNS record types. IPv4 and IPv6 are supported in DNS name-to-address and DNS address-to-name lookup processes. Specifying a name server and enabling DNS forwarding maps both IPv4 and IPv6 addresses.

IPv6 addresses

You can configure DNS Relay to use IPv6 addresses using the same commands used to configure DNS Relay to use IPv4 addresses, as shown in the following example:

```
awplus# configure terminal
awplus(config)# ip name-server 2001:0db8:010d::1
awplus(config)# ip name-server 2001:0db8:010d::2
awplus(config)# ip dns forwarding
```

DNS Relay commands

You can then configure DNS Relay behavior with the following commands:

To set the number of times a device will retry to forward DNS queries, use the command:

```
awplus(config)# ip dns forwarding retry <0-100>
```

To set the number of seconds to wait for a response, use the command:

```
awplus(config)# ip dns forwarding timeout <0-3600>
```

To set the DNS forwarding dead-time period in seconds, use the command:

```
awplus(config)# ip dns forwarding dead-time <60-43200>
```

At the dead-time period set, the device stops sending requests to an unresponsive server.

To set the interface to use for forwarding and receiving DNS queries, use the command:

```
awplus(config)# ip dns forwarding source-interface <interface-name>
```

To specify the DNS Relay name resolver cache size and lifetime, use the command:

```
awplus(config)# ip dns forwarding cache [size <0-1000>] [timeout <60-
3600>]
```

To remove entries from the DNS Relay name resolver cache, use the command:

```
awplus(config)# clear ip dns forwarding cache
```

Information which may be useful for troubleshooting DNS Relay is available using the DNS Relay debugging function. To enable DNS Relay debugging, use the command:

```
awplus# debug ip dns forwarding
```

To display the status of DNS Relay, use the command:

```
awplus# show ip dns forwarding
```

To display the status of DNS Relay name servers, use the command:

```
awplus# show ip dns forwarding server
```

To display the DNS Relay name resolver cache, use the command:

```
awplus# show ip dns forwarding cache
```

DNS operation with VRF-lite

From release 5.5.2-0.1 onwards, on devices that support VRF-lite, you can configure the DNS Client and DNS Relay functionality to be VRF aware.

In this mode the DNS Client will use name-servers configured for the VRF, and DNS Relay will forward DNS messages within specified VRF instances.

Configuring DNS operation with VRF-lite.

The **ip name-server [vrf *<name>*] *<ip-addr>*** command configures a name-server for the specified VRF. This command assigns the address of one or more name servers to a VRF table to be used for name and address resolution. If no VRF-lite instance (vrf<*name*>) is specified, the name-server is configured for the global VRF. A VRF specific name-cache is created within the DNS relay for every VRF instance that has a name-server configured.

A maximum of three name-servers may be defined for each VRF instance.

The configuration command, **ip dns forwarding**, will apply to all VRF instances configured on the device and not on a per VRF basis.

The configuration commands listed below apply to all VRF instances configured on the device and not on a per VRF basis. Timeouts are in seconds as per existing commands:

- `ip dns forwarding retry`
- `ip dns forwarding timeout`
- `ip dns forwarding dead-time`
- `ip dns forwarding source-interface`
- `ip dns forwarding cache`

The following **show** commands provide output information for the VRF instance specified. If a VRF instance is not specified, output is shown for all VRF instances, including the global instance and the output will be formatted in a way that distinguishes the information for each VRF.

- `show ip dns [vrf <name>|global] forwarding server`
- `show ip dns [vrf <name>|global] forwarding cache`
- `show ip name-server [vrf <name>|global]`

The DNS cache can also be cleared on a per VRF instance basis by using the **clear ip dns [vrf *<name>*|global] forwarding cache** command.

The DNS client can be made VRF-aware by forwarding all lookups to the DNS relay.

To configure the DNS client to be VRF-aware, use the commands:

Output

```
awplus# configure terminal
awplus(config)# ip domain-lookup via-relay
awplus(config)# ip dns forwarding
```

The following commands show how to configure a DNS relay name-server for both the specified VRF instance VRF red, and the global VRF instance.

To configure a DNS relay name-server for the VRF-lite instance red:

```
awplus# configure terminal
awplus(config)# ip name-server vrf red 192.168.0.1
awplus(config)# ip domain-lookup
```

To configure a DNS relay name-server for the global VRF instance:

```
awplus# configure terminal
awplus(config)# ip name-server 192.168.1.1
awplus(config)# ip domain-lookup
```

DNS name resolver caching

You can enable **DNS name resolver caching** on the DNS relay, which provides a lookup speed advantage, and avoids unnecessary repeated requests to external DNS servers.

When you enable DNS Relay name resolver cache, the device will maintain a cache of recently used mappings between domain names and IP addresses so that other identical requests can be responded to without further reference to an external, or upstream DNS server.

- The DNS cache has a limited size, and times out entries after a specified period of up to 60 minutes.
- DNS caching is disabled by default.

DHCP options

When your device is using its DHCP client for an interface, it can receive the following DHCP options from the DHCP server:

- Option 6 - a list of DNS servers. This list appends to the dynamic DNS server set on your device with the **ip name-server** command. If you want to change the preference to static, use the command **ip name-server preferred-order static**.
- Option 15 - a domain name used to resolve host names. This option replaces the domain name set with the **ip domain-name** command.

PPP options

When your device is using a PPP interface, it can configure to learn or negotiate DNS servers using the command **ppp ipcp dns**.

DNS domain name matching

DNS domain name matching allows you to specify a domain name suffix to match on when a client does a DNS lookup. When a match is detected the appropriate name server is used for resolving the address. For example, you might want to do the following:

- For all general DNS lookups such as www.example.com, use name server 1.2.3.4.
- For internal domain names such as atlnz.lc, use the name server received on interface ppp1.

Domain matching order

The matching procedure used for this feature is a tail-first match against the requested domain in the DNS request packet, and the longest specified domain will match first. The following examples show you how this works:

If the configuration is set up so that:

- ‘com’ will be sent to DNS server 10.1.1.1.
- ‘example.com’ will be sent to DNS server 20.1.1.1.
- otherwise everything else will be sent to the primary DNS server 30.1.1.1.

Then the following will occur:

1. When a DNS request for ‘mail.example.com’ is received, the longest successful match is ‘example.com’, so the request will be forwarded on to DNS server 20.1.1.1.
2. When a DNS request for ‘something.com’ is received, the longest successful match is against ‘com’, so the request will be forwarded on to DNS server 10.1.1.1.
3. When a DNS request for ‘something_else.org’ is received, there is no match against the specified domains, so the request will be forwarded on to the primary DNS server 30.1.1.1.

Note: This feature will only work for DNS lookups from a down-stream host that are relayed via the device, which is enabled by using the **ip dns forwarding** command. The feature functionality does not apply to DNS lookups originating from the device itself, for example ‘ping hostname’ executed on the device.

The following example shows you how to create a domain list that can be used as a suffix-list for DNS lookups. The domain list includes domains that are internal to the company such as ‘engineering.acme’ or ‘intranet.acme’.

To create a domain list called ‘corporatedomains’, use the command:

```
awplus(config)# ip dns forwarding domain-list corporatedomains
```

To give a description to the domain list, use the command:

```
awplus(config-domain-list)# description Our internal network domains
```

To add a domain to the list, use the command:

```
awplus(config-domain-list)# domain engineering.acme
awplus(config-domain-list)# domain intranet.acme
```

Dynamic Domain Name System (DDNS)

Dynamic Domain Name System (DDNS) is a mechanism which allows a DDNS client to automatically update a DNS entry hosted by a DDNS Provider. When DDNS is configured on an AR-Series Firewall, DNS requests are automatically directed to the configured host name regardless of Dynamic IP address changes. This feature is available on all AR-Series Firewalls from release 5.4.7 onwards.

Hosting your own web server normally requires a static IP address from your ISP to ensure that your services are always reachable. Your domain name maps your static IP address to your domain (via DNS). Home users or small offices may not want to pay for a static IP address so can use DDNS with a dynamic IP address instead.

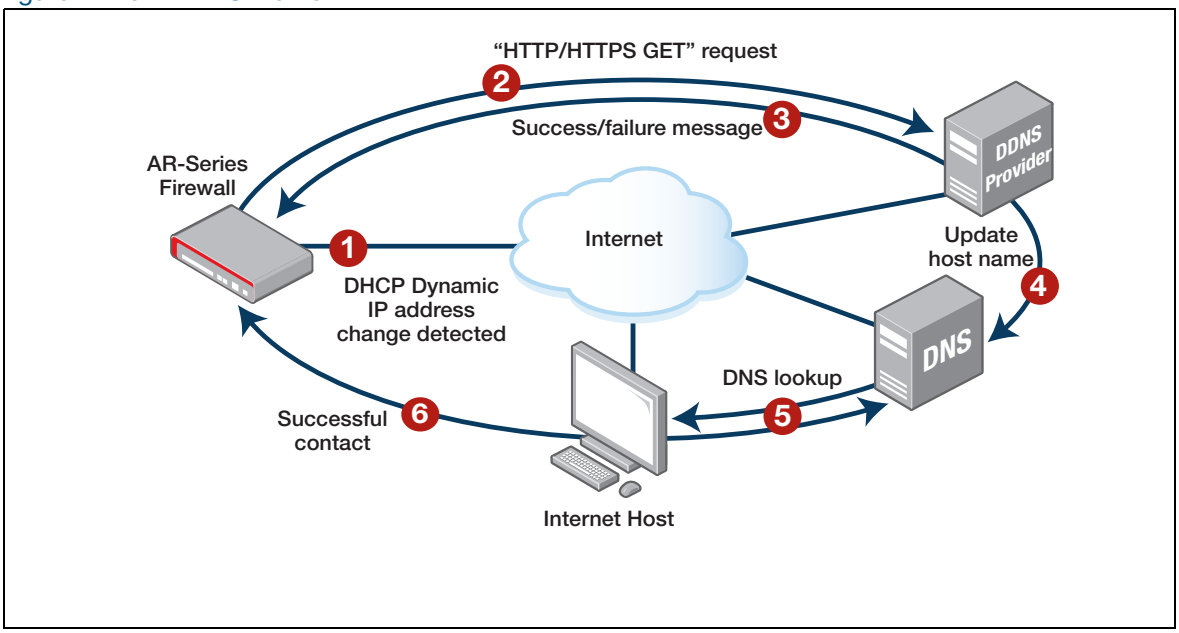
DDNS is a method of updating DNS records without the need for manual editing. DDNS provides a consistent domain name for a host that may have its global IP address changed at any time. The DDNS client updates the service providers records to keep the domain name pointing to the correct IP address.

How does DDNS work?

Typically an ISP assigns an **exit interface** on an edge AR-Series Firewall with a **dynamic public IP address** learned by DHCP. This can make establishing a connection over the Internet with the exit interface (and its local LAN) difficult because a dynamic public IP address can change without warning.

DDNS overcomes this by associating a **DNS host name** with the IP address on the exit interface. When the IP address on the exit interface changes, the device contacts the DDNS Provider who then updates the associated DNS host name with the new IP address. This means that any packets sent to the associated host name are sent to the device regardless of what the IP address is on the exit interface.

Figure 1: How DDNS works:



This feature works with the HTTP/HTTPS method of DDNS as follows:

1. The AR-Series Firewall detects a change in the IP address on an interface associated with a DDNS method
2. The AR-Series Firewall sends an 'HTTP/HTTPS GET' request to the configured DDNS Provider.
3. The DDNS Provider sends back an 'HTTP/HTTPS' message indicating a success or failure.
4. If the configured user credentials are correct, the DDNS Provider updates the associated DNS host name.
5. When the Internet Host wants to connect to the AR-Series Firewall the DNS lookup resolves the current IP address.
6. The Internet Host successfully makes contact.

Note: This version of DDNS does not support changing of MX (eMail eXchanger) records.

The following steps are required to configure DDNS:

1. **Register a DNS host name with a DDNS Provider**
2. **Create a DDNS method**
3. **Configure the update URL**
4. **Enter the required user name, password and host name**
5. **Apply a DDNS method to an exit interface**
6. **Configure the exit interface to learn an IP address dynamically**
7. **Enable DDNS**
8. **Check the status of configured update methods**

Each of these steps is described in detail as follows:

Step 1: **Register a DNS host name with a DDNS Provider**

Create an account with your DDNS Provider and then log in to your account so that you can set up your host name. When you register your host name, it will have a user name, password, and domain name assigned.

Examples of DDNS providers are:

- Dyn (formerly DynDNS)
- No-IP
- Dynu
- Duck DNS
- Dynv6

Step 2: **Create a DDNS method**

The **ddns-update-method** command is configured from Global Configuration mode. To enter DDNS Update Method Configuration mode and create a new method, use the command:

```
awplus(config)# ddns-update-method <method-name>
awplus(config-ddns-update-method)#
```

For example, to create a method named ‘dyndns’, use the command:

```
awplus(config)# ddns-update-method dyndns
awplus(config-ddns-update-method)#
```

Step 3: **Configure the update URL**

The **update-url** command is configured from DDNS Update Method Configuration mode. To configure the update URL, use the command:

```
awplus(config-ddns-update-method)# update-url <url-name>
```

The update URL (provided by the DDNS Provider) can include a user name, password, host name and/or IP address. These user values are optional because they may vary depending on the DDNS Providers update URLs. AlliedWare Plus requires you to enter the required parameters for the update URL using the following placeholder tokens:

For the placeholder token:

- user name, enter *<user-name>*
- password, enter *<password>*
- host name, enter *<host-name>*
- IP address, enter *<ip-address>*

For example, for Dyn DNS the following update URL can be used:

http://username:password@members.dyndns.org/nic/update?SYSTEM=dyndns&hostname=<h>&myip=<a>

To configure this URL, use the following command including the placeholder tokens:

```
awplus(config-ddns-update-method)# update-url http://
<USERNAME>:<PASSWORD>@members.dyndns.org/nic/
update?SYSTEM=dyndns&hostname=<HOST-NAME>&myip=<IP-ADDRESS>
```

Note: Dyn DNS also has the following update URL that can be used: http://<USERNAME>:<PASSWORD>@members.dyndns.org/v3/update?hostname=<HOST-NAME>&myip=<IPADDRESS>

See **Securing DDNS updates** for examples using this update URL.

URLs that contain the character '?' activate help from the command line. To stop the help from activating enter the '?' in the command line, then press Ctrl+v.

Step 4: Enter the required user name, password and host name

Depending on the DDNS Provider, use the commands **username**, **password** and **host-name** to enter the registered details associated with the host name. The value for the placeholder token <IPADDRESS> is populated automatically from the interface IP settings. These commands are entered from DDNS Update Method Configuration mode.

To configure a user name, use the command:

```
awplus(config-ddns-update-method)# username <user-name>
```

To configure a password, use the command:

```
awplus(config-ddns-update-method)# password <password>
```

To configure a host name, use the command:

```
awplus(config-ddns-update-method)# host-name <host-name>
```

For example, the update URL for Dyn DNS requires a user name, password and host name to be entered as follows:

```
awplus(config-ddns-update-method)# username atlnz
awplus(config-ddns-update-method)# password
20zwkLPCEKk9tcE55yXE2UJH8lBP65h0
awplus(config-ddns-update-method)# host-name example-atlnz.dyndns.org
```

Step 5: Apply a DDNS method to an exit interface

DDNS can be applied to interface types PPP, eth and VLAN. These commands are entered from Interface Configuration mode.

To apply a DDNS method to an IPv4 interface, use the command:

```
awplus(config-if)# ip ddns-update-method <method-name>
```

To apply a DDNS method to an IPv6 interface, use the command:

```
awplus(config-if)# ipv6 ddns-update-method <method-name>
```

For example, to apply the DDNS method ‘dyndns’ to an IPv4 address on interface eth1, use the commands:

```
awplus(config)# interface eth1
awplus(config-if)# ip ddns-update-method dyndns
```

Step 6: **Configure the exit interface to learn an IP address dynamically**

You can configure your device with DHCP or PPP from Interface Configuration mode.

For example, to configure exit interface eth1 with an IPv4 address using DHCP, use the command:

```
awplus(config-if)# ip address dhcp
```

Step 7: **Enable DDNS**

DDNS updates are disabled by default. The command **ddns enable** is configured from Global Configuration mode. To globally enable DDNS updates, use the commands:

```
awplus# configure terminal
awplus(config)# ddns enable
```

Step 8: **Check the status of configured update methods**

To show the status of configured DDNS update methods, from either Privileged Exec or User Exec mode, use the command:

```
awplus# show ddns-update-method status
```

Basic configuration example

A DDNS method named ‘dyndns’ has been created using the DDNS Provider Dyn DNS. This method has been applied to the dynamically configured DHCP IPv4 address on the exit interface eth1.

```
awplus# configure terminal
awplus(config)# ddns-update-method dyndns
awplus(config-ddns-update-method)# update-url http://
<USERNAME>:<PASSWORD>@members.dyndns.org/nic/
update?SYSTEM=dyndns&hostname=<HOST-NAME>&myip=<IPADDRESS>
awplus(config-ddns-update-method)# username atlnz
awplus(config-ddns-update-method)# password
20zwkLPCEKk9tcE55yXE2UJH8lBP65h0
awplus(config-ddns-update-method)# host-name example-atlnz.dyndns.org
awplus(config-ddns-update-method)# exit
awplus(config)# interface eth1
awplus(config-if)# ip ddns-update-method dyndns
awplus(config-if)# ip address dhcp
awplus(config-if)# exit
```

```
awplus(config)# ddns enable
```

Basic configuration example:

```
ddns enable
!
ddns-update-method dyndns
 update-url http://<USERNAME>:<PASSWORD>@members.dyndns.org/nic/
      update?SYSTEM=dyndns&hostname=<HOST-NAME>&myip=<IPADDRESS>
 username atlnz
 password 20zwkLPCEKk9tcE55yXE2UJH81BP65h0
 host-name example-atlnz.dyndns.org
!
interface eth1
 ip ddns-update-method dyndns
 ip address dhcp
```

The **show ddns status** command displays the status of currently configured DDNS update method:

```
awplus#show ddns status

Dynamic DNS updates are enabled

------------------------------------------------------------------------
Update Method Name              dyndns
Hostname                        example-atlnz.dyndns.org
IPv4 Interface                  eth1
IPv4 Address                    203.0.113.123
IPv4 Status                     IPv4 update succeeded
IPv4 Update Result              good 203.0.113.123
IPv6 Interface                  None
IPv6 Address                    None
IPv6 Status                     None
IPv6 Update Result              None
Last update                     2017-03-16T22:32:43Z
```

Securing DDNS updates

URL updates can be sent securely via HTTPS/SSL. DDNS updates can also be sent to a DDNS provider via a variety of port numbers. This can be useful to bypass HTTP proxies which could potentially edit HTTP communications between the client and server. If you are configuring a specific port, DDNS updates will only succeed if the DDNS provider supports the port that you have specified.

To add an update URL to the DDNS update method ‘dyndns’ that uses HTTP, use the commands:

```
awplus(config)# ddns-update-method dyndns
awplus(config-ddns-update-method)# update-url http://
<USERNAME>:<PASSWORD>@members.dyndns.org/v3/update?hostname=
<HOST-NAME>&myip=<IPADDRESS>
```

To add an update URL to the DDNS update method ‘dyndns’ that uses HTTPS/SSL, use the commands:

```
awplus(config)# ddns-update-method dyndns
awplus(config-ddns-update-method)# update-url https://
<USERNAME>:<PASSWORD>@members.dyndns.org/v3/update?hostname=
<HOST-NAME>&myip=<IPADDRESS>
```

To add an update URL to the DDNS update method ‘dyndns’, that uses HTTP on port 8245, use the commands:

```
awplus(config)# ddns-update-method dyndns
awplus(config-ddns-update-method)# update-url http://
<USERNAME>:<PASSWORD>@members.dyndns.org:8245/v3/update?hostname=<HOST-
NAME>&myip=<IPADDRESS>
```

Optional DDNS commands

Update interval

By default the device only sends DDNS updates to the DDNS service provider when a change in the IP address triggers an update. The **update-interval** command can specify a time interval for when periodic updates are sent to the provider. The update interval is from 1 minute to 45 days expressed in minutes. From DDNS Update Method mode, to specify the time interval between periodic DDNS updates, use the command:

```
awplus(config-ddns-update-method)# update-interval <1-64800>
```

For example, to enable periodic DDNS updates every day for the method ‘dyndns’, use the commands:

```
awplus(config)# ddns-update-method dyndns
awplus(config-ddns-update-method)# update-interval 1440
```

To disable periodic DDNS updates for the method ‘dyndns’, use the commands:

```
awplus(config)# ddns-update-method dyndns
awplus(config-ddns-update-method)# no update-interval
```

Retry interval

By default no retry intervals are set. The **retry-interval** command is used to enable DDNS update retries. Retries are attempted after a DDNS update fails after the specified interval. The retry interval is from 1 second to 4.5 days expressed in seconds. If the DDNS update keeps failing then no more than the specified number of maximum retries are attempted. Maximum retries range from 1 to 100 times. An update triggered by another source, such as an IP address change, or a manual update will reset the retry counter to begin from the start.

The retry interval is used for one DDNS update at one time, so if an update is not complete within the specified interval, an update will not begin until it has completed.

From DDNS Update Method mode, to enable retries, use the command:

```
awplus(config-ddns-update-method)# retry-interval <1-3888000> maximum-
retries <1-100>
```

For example, to enable DDNS update retry attempts for the method ‘dyndns’ every hour up to 5 times, use the commands:

```
awplus(config)# ddns-update-method dyndns
awplus(config-ddns-update-method)# retry-interval 300 maximum-retries 5
```

For example, to disable DDNS update retry attempts for the method ‘dyndns’, use the commands:

```
awplus(config)# ddns-update-method dyndns
awplus(config-ddns-update-method)# no retry-interval
```

Manual updates for DDNS methods

The command **ddns-update now** is used to manually update DDNS methods. When no method name is entered, all DDNS update methods are updated. When an update method is specified, then only the host name configured for that method is updated. From Privileged Exec mode, to manually update all DDNS methods, use the command:

```
awplus# ddns-update now
```

To specify a method, use the command:

```
awplus# ddns-update method <method-name> now
```

For example, to manually update only the DDNS update method ‘dyndns’, use the command:

```
awplus# ddns-update dyndns now
```

Debug commands

From Priviledged Exec mode, use the **debug** command to see what is being sent via HTTP/HTTPS.

To enable debugging of the DDNS process, use the command:

```
awplus# debug ddns
```

To disable debugging of the DDNS process, use the command:

```
awplus# no debug ddns
```

or

```
awplus# undebug ddns
```

To display information for all debugging options, use the command:

```
awplus# show debugging ddns
```

IPv6 over IPv4 updates

By default IPv6 updates are sent using IPv6. However, if your DDNS provider supports IPv6 but does not allow sending updates in IPv6, then you can use the command **use-ipv4-for-ipv6 updates** to send updates for IPv6 addresses transported using IPv4 instead. From DDNS Update Method Configuration mode to send IPv6 updates using IPv4, use the command:

```
awplus(config-ddns-update-method)# use-ipv4-for-ipv6 updates
```

To stop sending IPv6 updates using IPv4, use the commands:

```
awplus(config-ddns-update-method)# no use-ipv4-for-ipv6 updates
```

The **suppress-ipv4-updates** command stops IPv4 DDNS updates from being sent and sends only IPv6 updates. This command is used in conjunction with the **use-ipv4-for-ipv6 updates** command:

```
awplus(config-ddns-update-method)# suppress-ipv4-updates
```

For example, Dyn DNS does not send updates in IPv6, so use these commands to send updates using IPv4 instead, and stop IPv4 updates from being sent:

```
awplus(config)# ddns-update-method dyndns
awplus(config-ddns-update-method)# use-ipv4-for-ipv6 updates
awplus(config-ddns-update-method)# suppress-ipv4-updates
```

DDNS via NAT configuration example

If a NAT device exists between the AR-Series Firewall and the DDNS Provider then depending on the DDNS provider and/or the configured update URL, the DNS host name can be updated with the IP address of the NAT device, rather than that of the AR-Series Firewall.

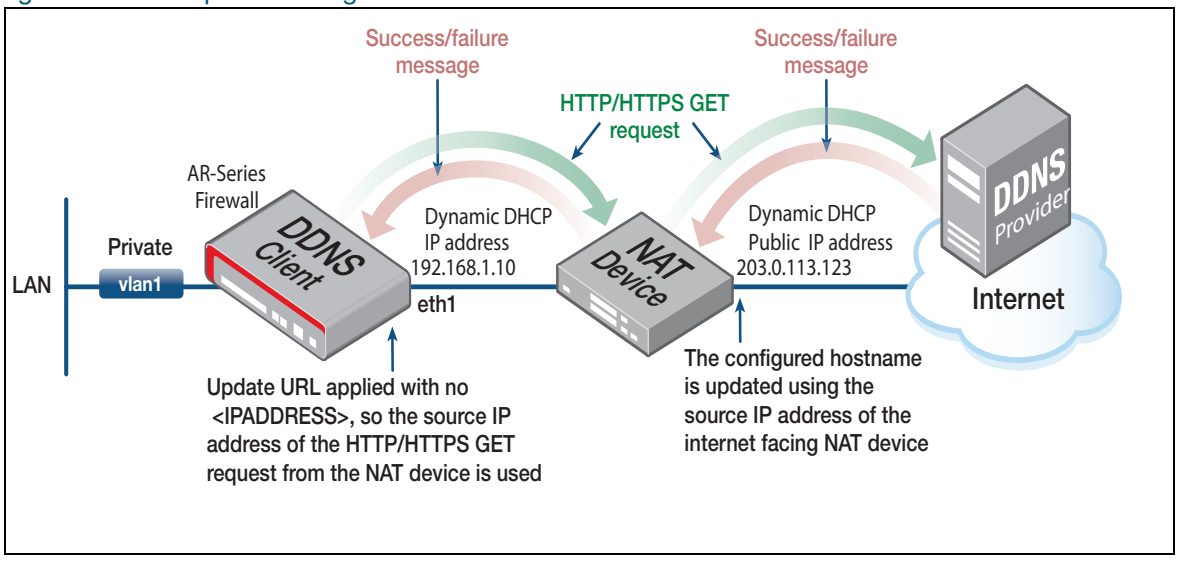
This occurs if the DDNS Provider uses the **source IP address** of the 'HTTP/HTTPS GET' request.

- Some DDNS providers (Dyn DNS, no-ip) will use the source IP address if an IP address is not specified in the update URL.
- Other DDNS providers (dynu) do not allow an IP address to be present in the update URL so can only use the source IP address.

In this configuration example using the DDNS Provider Dyn DNS, the <IPADDRESS> placeholder token is not configured in the update URL. Dyn DNS will use the source IP address of the 'HTTP/HTTPS GET' request from the Internet facing NAT device.

By default updates are sent when the IP address changes on the AR-Series Firewall where the DDNS update method is attached. In this case we want an update sent when the IP address on the NAT device changes, not when the IP address on the AR-Series Firewall changes. An update interval is configured so that when the IP address on the NAT device changes, the associated hostname will be updated within 20 minutes of the change.

Figure 2: DDNS updates using the source IP address of the NAT device:



DDNS via NAT configuration example:

```
 update-url http://<USERNAME>:<PASSWORD>@members.dyndns.org/nic/
                        update?SYSTEM=dyndns&hostname=<HOST-NAME>
 host-name example-atlnz.dyndns.org
 username atlnz
 password 20zwkLPCEKk9tcE55yXE2UJH81BP65h0
  update-interval 20
!
interface eth1
 ip ddns-update-method dyndns
 ip address dhcp
```

Interoperability with a firewall configuration example

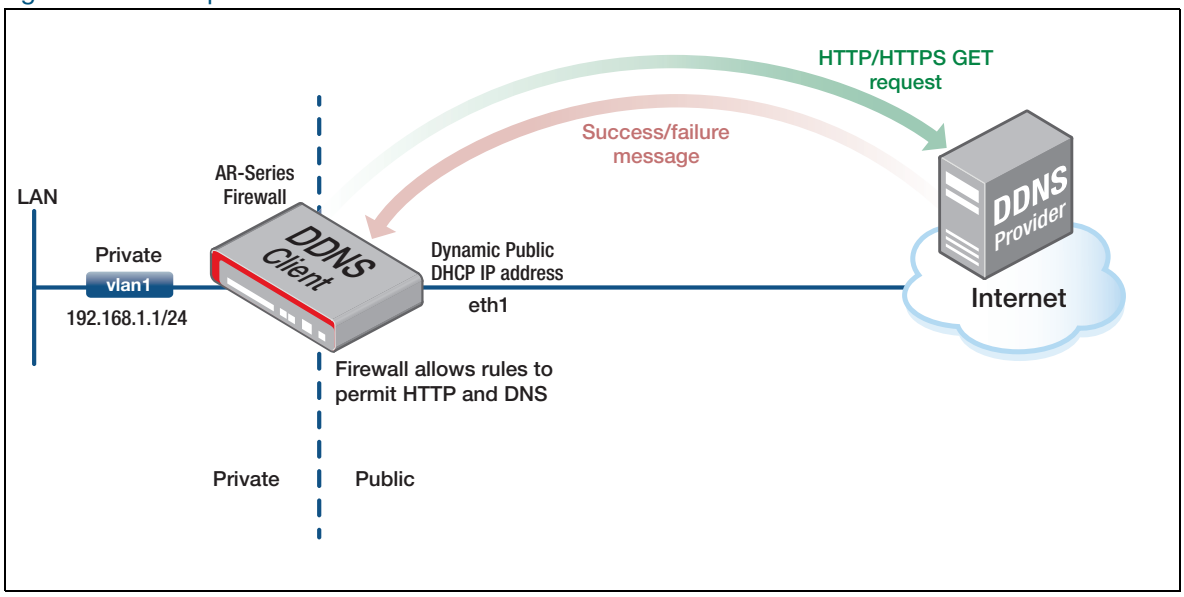
The following configuration example allows DDNS to inter-operate with a firewall. The private zone defines the LAN which is connected to vlan1. The public zone defines the WAN for the Internet which is connected to eth1 and has a dynamic IP address configured.

The following firewall rules have been created:

- Rule 10 allows any connections initiated from the LAN to the Internet.
- Rule 20 allows HTTP traffic from the AR series firewall to the Internet. Rule 20 is necessary because in this example the DDNS method is configured to use HTTP for DDNS updates. If HTTPS is used DDNS updates rather than HTTP, then rule 20 would need to reflect this.
- Rule 30 allows DNS traffic from the AR series firewall to the Internet. Rule 30 is necessary because DDNS uses DNS to resolve the DDNS Provider as described in the update URL.

The DDNS method is applied to the exit interface eth1 of the AR-Series Firewall and has a dynamically assigned DHCP IP address configured. DDNS updates will be sent when IP address changes occur.

Figure 3: DDNS updates from a firewall:



DDNS updates from a firewall configuration example:

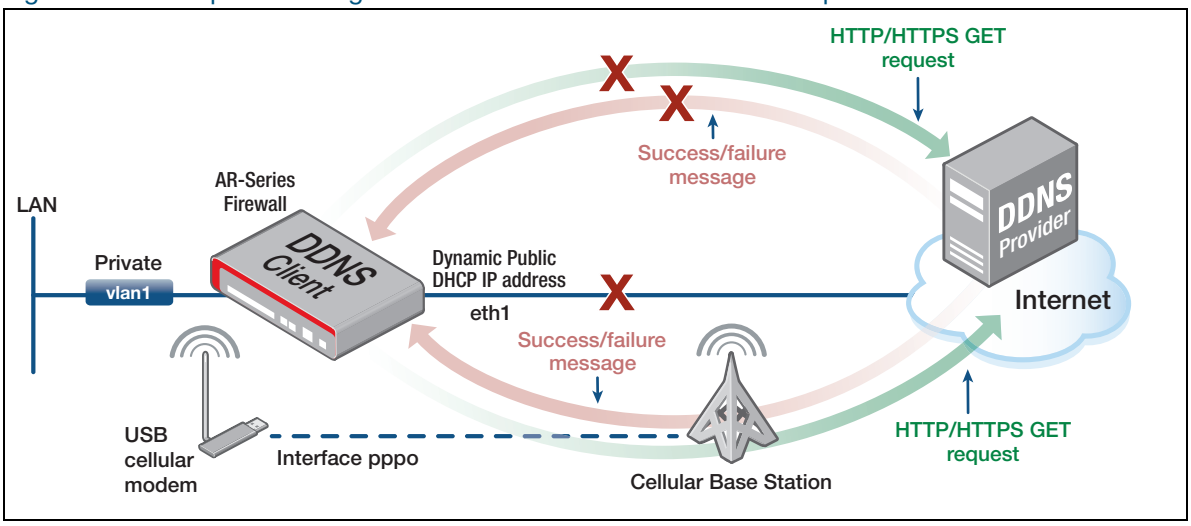
```
interface eth1
 ip ddns-update-method dyndns
 ip address dhcp
!
interface vlan1
 ip address 192.168.1.1/24
!
ddns-update-method dyndns
 update-url http://<USERNAME>:<PASSWORD>@members.dyndns.org/nic/
      update?SYSTEM=dyndns&hostname=<HOST-NAME>&myip=<IPADDRESS>
 host-name example-atlnz.dyndns.org
 username atlnz
 password 20zwkLPCEKk9tcE55yXE2UJH81BP65h0
!
zone private
 network lan
  ip subnet 192.168.1.0/24 interface vlan1
!
zone public
 network wan
  ip subnet 0.0.0.0/0 interface eth1
  host interface
   ip address dynamic interface eth1
!
firewall
 rule 10 permit any from private to public
 rule 20 permit http from public.wan.interface to public
 rule 30 permit dns from public.wan.interface to public
 protect
```

3G USB Cellular Modem backup interface configuration example

The following configuration example uses a 3G USB Cellular Modem as a backup link to the Internet.

If the main eth1 link fails, then a backup link can be configured via a USB Cellular Modem. In this configuration example, a 3G Cellular Modem (interface cellular0) has been configured using ppp0. A script is triggered to change the DDNS method to be assigned to the ppp0 interface if eth1 is down. Another script is triggered to change the DDNS method to be assigned back to eth1 when it is up again.

Figure 4: DDNS updates using a 3G USB Cellular Modem as a backup when eth1 fails:



When eth1 is up it is used for Internet connectivity and the DDNS method will send DDNS updates when IP address changes occur on the AR-Series Firewall.

When eth1 goes down, the trigger script **unreachable.scp** is activated. This script deletes the DDNS method from eth1 and assigns it to the ppp0 interface (cellular0) associated with the 3G USB Cellular Modem. This script then enables the ppp0 interface. The ppp0 interface (cellular0) is now used for Internet connectivity and the DDNS method will send DDNS updates when IP address changes occur on the AR-Series Firewall via the 3G USB Cellular Modem.

When eth1 comes back up, the script **reachable.scp** is triggered which deletes the DDNS method from the ppp0 interface (cellular0), shuts down the ppp0 interface and assigns the DDNS method back to eth1. The DDNS method will resume sending DDNS updates when IP address changes occur on the AR-Series Firewall.

3G USB Cellular Modem as a backup link with DDNS configuration example:

```
ddns-update-method dyndns
 update-url http://<USERNAME>:<PASSWORD>@members.dyndns.org/nic/
      update?SYSTEM=dyndns&hostname=<HOST-NAME>&myip=<IPADDRESS>
 host-name example-atlnz.dyndns.org
 username atlnz
 password 20zwkLPCEKk9tcE55yXE2UJH81BP65h0
!
interface eth1
 ip ddns-update-method dyndns
 ip address dhcp
!
interface vlan1
 ip address 198.18.8.2/24
!
interface cellular0
 encapsulation ppp 0
 apn internet
!
interface ppp0
 ppp ipcp dns request
 keepalive
 ip address negotiated
 shutdown
 ip tcp adjust-mss pmtu
!
trigger 1
 type interface eth1 down
 script 1 unreachable.scp
trigger 2
 type interface eth1 up
 script 1 reachable.scp
```

If eth1 is down, then this script switches the DDNS method over to the backup cellular interface. Trigger 1 unreachable script:

```
enable
conf t
interface eth1
no ip ddns-update-method dyndns
interface ppp0
ip ddns-update-method dyndns
no shut
```

When eth1 comes back up, then this script switches the DDNS method back to the eth1 exit interface. Trigger 2 reachable script:

```
enable
conf t
interface ppp0
shut
no ip ddns-update-method dyndns
interface eth1
ip ddns-update-method dyndns
```

DDNS with Dynamic Peer to Peer VPN configuration example

The following configuration can be used to establish an IPsec connection between two AR-Series Firewalls with Dynamic IP addresses. When the IPsec tunnel destination is configured as a host name it will resolve the host name to determine the destination IP address.

With the DDNS client configured in each AR-Series Firewall a change of dynamically allocated WAN address will cause the AR-Series Firewall to send an update to the DDNS provider.

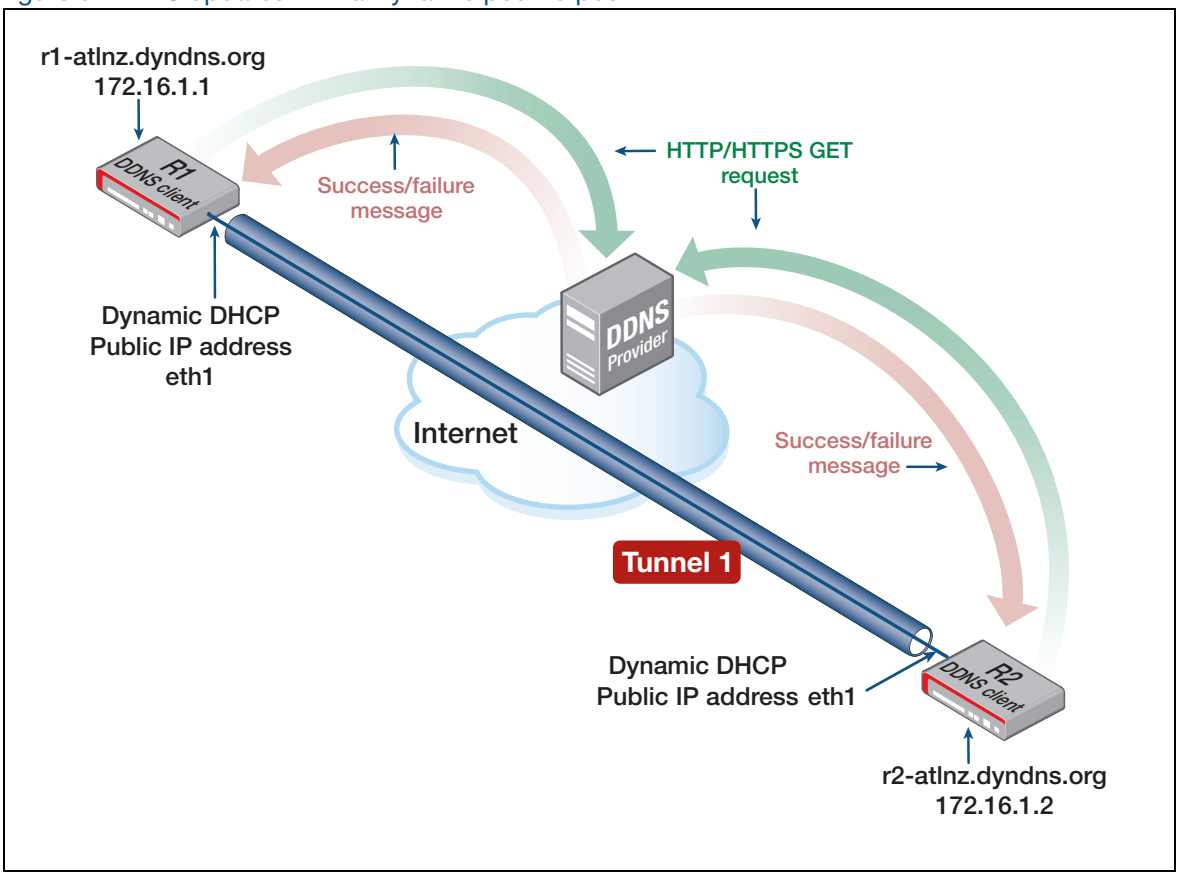
The peer located at the remote site of the IPsec VPN will initially not be aware of this change and will continue to send tunnel traffic to the originally resolved IP address. After a period of time however, dead peer detection (DPD) will detect that the remote peer is no longer reachable, at which point the host-name associated with the remote peer is re-resolved using DNS and the tunnel is renegotiated using the new IP address.

Each of the routers has a host name associated with them which is updated via DDNS.

- Router 1 is associated with the host name **r1-atlnz.dyndns.org**.
- Router 2 is associated with the host name **r2-atlnz.dyndns.org**.

There is an IPsec tunnel between Router 1 and Router 2. The IPsec tunnel uses the associated host name of the remote router as its destination.

Figure 5: DDNS updates with a Dynamic peer to peer VPN:



Router 1 Configuration example:

```
crypto isakmp key friend hostname r1-atlnz.dyndns.org
crypto isakmp key friend hostname r2-atlnz.dyndns.org
!
ddns enable
!
ddns-update-method dyndns
update-url http://<USERNAME>:<PASSWORD>@members.dyndns.org/nic/
update?SYSTEM=dyndns&hostname=<HOST-NAME>&myip=<IPADDRESS>
host-name r1-atlnz.dyndns.org
username atlnz
password 20zwkLPCEKk9tcE55yXE2UJH81BP65h0
retry-interval 1 maximum-retries 5
!
interface eth1
ip ddns-update-method dyndns
ip address dhcp
!
interface tunnel1
mtu 1438
tunnel source eth1
tunnel destination r2-atlnz.dyndns.org
tunnel local name r1-atlnz.dyndns.org
tunnel remote name r2-atlnz.dyndns.org
tunnel protection ipsec
tunnel mode ipsec ipv4
ip address 172.16.1.1/30
!
```

Router 2 Configuration example:

```
!
crypto isakmp key friend hostname r1-atlnz.dyndns.org
crypto isakmp key friend hostname r2-atlnz.dyndns.org
ddns enable
!
ddns-update-method dyndns
update-url http://<USERNAME>:<PASSWORD>@members.dyndns.org/nic/
update?SYSTEM=dyndns&hostname=<HOST-NAME>&myip=<IPADDRESS>
host-name r2-atlnz.dyndns.org
username atlnz
password 20zwkLPCEKk9tcE55yXE2UJH81BP65h0
retry-interval 1 maximum-retries 5
!
interface eth1
ip ddns-update-method dyndns
ip address dhcp
!
interface tunnel1
mtu 1438
tunnel source eth1
tunnel destination r1-atlnz.dyndns.org
tunnel local name r2-atlnz.dyndns.org
tunnel remote name r1-atlnz.dyndns.org
tunnel protection ipsec
tunnel mode ipsec ipv4
ip address 172.16.1.2/30
!
```

For information on configuring this example before AlliedWare Plus software version 5.5.2-0.1, see the "Appendix:" on page 28.

Appendix:

DDNS with Dynamic Peer to Peer VPN configuration example - before version 5.5.2-0.1

Before AlliedWare Plus version 5.5.2-0.1, the following configuration can be used to establish an IPsec connection between two AR-Series Firewalls with Dynamic IP addresses. When the IPsec tunnel destination is configured as a host name it will resolve the host name to determine the destination IP address.

With the DDNS client configured in each AR-Series Firewall a change of dynamically allocated WAN address will cause the AR-Series Firewall to send an update to the DDNS provider. However, the peer located at the remote site of the IPsec VPN will not be aware of this change, and so the IPsec VPN will fail.

In order to speed up the detection of the IPsec VPN failure and recovery, ping poll is used to monitor the reachability of the IPsec peer, by pinging the IP address of the remote end of the IPsec VPN. If the IP address at the remote end of the IPsec tunnel becomes unreachable, then ping poll detects this loss of connectivity and activates trigger scripts which clears the broken IPsec VPN. This allows it to attempt to renegotiate to the new IP peer address via DNS lookup.

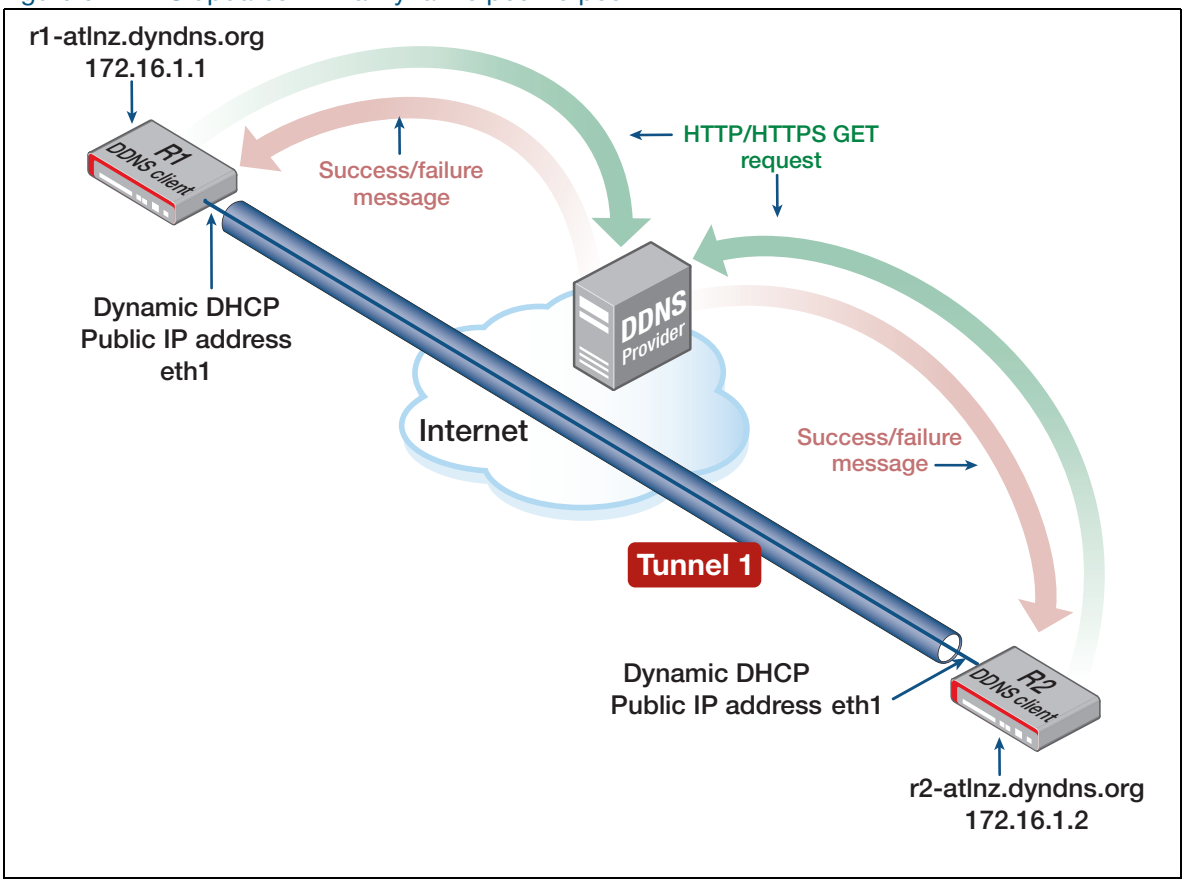
Each of the routers has a host name associated with them which is updated via DDNS.

- Router 1 is associated with the host name **r1-atlnz.dyndns.org**.
- Router 2 is associated with the host name **r2-atlnz.dyndns.org**.

There is an IPsec tunnel between Router 1 and Router 2. The IPsec tunnel uses the associated host name of the remote router as its destination. Each of the devices is configured with a ping poll which pings the IPsec virtual tunnel interface of the remote router via the IPsec VPN. If the public WAN IP address changes on one of the Routers:

1. The ping polls will fail and the device which had the IP address change will send a DDNS update.
2. When the ping polls fail, it will trigger the scripts **periodic-update-[1/2].scp**. These scripts configure periodic triggers with a period of one minute.
3. Every minute these periodic triggers will trigger the script **Host-change-[1/2].scp**.
4. The **Host-change-[1/2].scp** reconfigures the destination of the IPsec tunnel, forcing the tunnel to re-resolve the destination IP address via DNS lookup. This should allow the tunnel to reestablish, so each of the ping polls should then succeed.
5. When the ping poll succeeds, scripts **periodic-update-delete-[1/2].scp** will be run. These scripts delete the periodic triggers configured by **periodic-update-[1/2].scp**.

Figure 6: DDNS updates with a Dynamic peer to peer VPN:



Router 1 Configuration example:

```
ping-poll 1
 ip 172.16.1.2
 fail-count 3
 sample-size 10
 active
!
trigger 1
 type ping-poll 1 down
 script 1 periodic-update-1.scp
trigger 2
 type ping-poll 1 up
 script 1 periodic-update-delete-1.scp
!
crypto isakmp key friend hostname r1-atlnz.dyndns.org
crypto isakmp key friend hostname r2-atlnz.dyndns.org
!
!
!
ddns enable
!
ddns-update-method dyndns
 update-url http://<USERNAME>:<PASSWORD>@members.dyndns.org/nic/
update?SYSTEM=dyndns&hostname=<HOST-NAME>&myip=<IPADDRESS>
 host-name r1-atlnz.dyndns.org
 username atlnz
 password 20zwkLPCEKk9tcE55yXE2UJH81BP65h0
 retry-interval 1 maximum-retries 5
!
interface eth1
 ip ddns-update-method dyndns
 ip address dhcp
!
interface tunnel1
 mtu 1438
 tunnel source eth1
 tunnel destination r2-atlnz.dyndns.org
 tunnel local name r1-atlnz.dyndns.org
 tunnel remote name r2-atlnz.dyndns.org
 tunnel protection ipsec
 tunnel mode ipsec ipv4
 ip address 172.16.1.1/30
!
```

Trigger 1: periodic-update-1.scp:

```
enable
conf t
trigger 10
 type periodic 1
 script 1 Host-change-1.scp
```

Script 1 Host-change-1.scp:

```
enable
clear crypto isakmp sa force
conf t
int tunnel1
tunnel dest r2-atlnz.dyndns.org
```

Trigger 2 periodic-update-delete-1.scp:

```
enable
conf t
no trigger 10
```

Router 2 Configuration:

```
!
crypto isakmp key friend hostname r1-atlnz.dyndns.org
crypto isakmp key friend hostname r2-atlnz.dyndns.org
!
ping-poll 1
 ip 172.16.1.1
 up-count 3
 fail-count 3
 active
!
trigger 1
 type ping-poll 1 down
 script 1 periodic-update-2.scp
trigger 2
 type ping-poll 1 up
 script 1 periodic-update-delete-2.scp
!
ddns enable
!
ddns-update-method dyndns
 update-url http://<USERNAME>:<PASSWORD>@members.dyndns.org/nic/
update?SYSTEM=dyndns&hostname=<HOST-NAME>&myip=<IPADDRESS>
 host-name r2-atlnz.dyndns.org
 username atlnz
 password 20zwkLPCEKk9tcE55yXE2UJH81BP65h0
 retry-interval 1 maximum-retries 5
!
interface eth1
 ip ddns-update-method dyndns
 ip address dhcp
!
interface tunnel1
 mtu 1438
 tunnel source eth1
 tunnel destination r1-atlnz.dyndns.org
 tunnel local name r2-atlnz.dyndns.org
 tunnel remote name r1-atlnz.dyndns.org
 tunnel protection ipsec
 tunnel mode ipsec ipv4
 ip address 172.16.1.2/30
!
```

Trigger 1 periodic-update-2.scp:

```
enable
conf t
trigger 10
 type periodic 1
 script 1 Host-change-2.scp
```

Trigger 1 host-change-2.scp:

```
enable
clear crypto isakmp sa force
conf t
int tunnel1
tunnel dest r1-atlnz.dyndns.org
```

Trigger 2 periodic-update-delete-2.scp:

```
enable
conf t
no trigger 10
```

Show command to monitor the status of DDNS:

```
awplus#show ddns status

Dynamic DNS updates are enabled

------------------------------------------------------------------------
Update Method Name              dyndns
Hostname                        r1-atlnz.dyndns.org
IPv4 Interface                  eth1
IPv4 Address                    10.33.14.27
IPv4 Status                     Update succeeded
IPv4 Update Result              good 10.33.14.27
IPv6 Interface                  None
IPv6 Address                    None
IPv6 Status                     None
IPv6 Update Result              None
Last update                     2017-03-16T22.32:43Z
```


Allied Telesis NETWORK SMARTER

North America Headquarters | 19800 North Creek Parkway | Suite 100 | Bothell | WA 98011 | USA | T: +1 800 424 4284 | F: +1 425 481 3895
Asia-Pacific Headquarters | 11 Tai Seng Link | Singapore | 534182 | T: +65 6383 3832 | F: +65 6383 3830
EMEA & CSA Operations | Incheonweg 7 | 1437 EK Rozenburg | The Netherlands | T: +31 20 7950020 | F: +31 20 7950021

alliedtelesis.com

© 2024 Allied Telesis, Inc. All rights reserved. Information in this document is subject to change without notice. All company names, logos, and product designs that are trademarks or registered trademarks are the property of their respective owners.