TRAM
Internet-Draft
Intended status: Standards Track
Expires: February 24, 2017

P. Patil
T. Reddy
D. Wing
Cisco

August 23, 2016

TURN Server Auto Discovery
draft-ietf-tram-turn-server-discovery-09

Abstract

Current Traversal Using Relays around NAT (TURN) server discovery mechanisms are relatively static and limited to explicit configuration. These are usually under the administrative control of the application or TURN service provider, and not the enterprise, ISP, or the network in which the client is located. Enterprises and ISPs wishing to provide their own TURN servers need auto discovery mechanisms that a TURN client could use with no or minimal configuration. This document describes three such mechanisms for TURN server discovery.

Status of This Memo

This Internet-Draft is submitted in full conformance with the provisions of BCP 78 and BCP 79.

Internet-Drafts are working documents of the Internet Engineering Task Force (IETF). Note that other groups may also distribute working documents as Internet-Drafts. The list of current Internet-Drafts is at http://datatracker.ietf.org/drafts/current/.

Internet-Drafts are draft documents valid for a maximum of six months and may be updated, replaced, or obsoleted by other documents at any time. It is inappropriate to use Internet-Drafts as reference material or to cite them other than as "work in progress."

This Internet-Draft will expire on February 24, 2017.

Table of Contents

1. Introduction

TURN [RFC5766] is a protocol that is often used to improve the connectivity of Peer-to-Peer (P2P) applications (as defined in section 2.7 of [RFC5128]). TURN allows a connection to be established when one or both sides are incapable of a direct P2P connection. It is an important building block for interactive, real-time communication using audio, video, collaboration etc.

While TURN services are extensively used today, the means to auto discover TURN servers do not exist. TURN clients are usually explicitly configured with a well known TURN server. To allow TURN applications to operate seamlessly across different types of networks and encourage the use of TURN without the need for manual configuration, it is important that there exists an auto discovery mechanism for TURN services. Web Real-Time Communication (WebRTC) [I-D.ietf-rtcweb-overview] usages and related extensions, which are mostly based on web applications, need TURN server discovery mechanisms.

This document describes three discovery mechanisms, so as to maximize opportunity for discovery, based on the network in which the TURN client finds itself. The three discovery mechanisms are:

- A resolution mechanism based on straightforward Naming Authority Pointer (S-NAPTR) resource records in the Domain Name System (DNS). [RFC5928] describes details on retrieving a list of server transport addresses from DNS that can be used to create a TURN allocation.

- DNS Service Discovery

- A mechanism based on anycast address for TURN.

In general, if a client wishes to communicate using one of its interfaces using a specific IP address family, it SHOULD query the TURN server(s) that has been discovered for that specific interface and address family. How to select an interface and IP address family is out of the scope of this document.

2. Terminology

The key words "MUST", "MUST NOT", "REQUIRED", "SHALL", "SHALL NOT", "SHOULD", "SHOULD NOT", "RECOMMENDED", "NOT RECOMMENDED", "MAY", and "OPTIONAL" in this document are to be interpreted as described in [RFC2119].

3. Discovery Procedure

TURN clients, by default, discover TURN server(s) by means of local or manual TURN configuration (i.e., TURN servers configured at the system level). For example, in case of Web Real-Time Communication (WebRTC) [I-D.ietf-rtcweb-overview] usages and related extensions, which are based on web applications, a Java Script specified TURN server MUST be considered as local configuration. An implementation MAY give the user an opportunity (e.g., by means of configuration file options or menu items) to specify a TURN server for each address

family. A client can choose auto-discovery in the absence of local configuration, if local configuration doesn't work or in addition to local configuration. This document does not offer a recommendation on server selection.

A TURN client that implements the auto discovery algorithm, to discover TURN servers in the attached network, uses the following mechanisms for discovery:

- Service Resolution : The TURN client attempts to perform TURN service resolution using the host's DNS domain.

- DNS SD: DNS Service Discovery.

- Anycast : Send TURN allocate request to the assigned TURN anycast request for each combination of interface and address family.

Not all TURN servers may be discovered using NAPTR records or DNS SD; Similarly, not all TURN servers may support anycast. For best results, a client SHOULD implement all discovery mechanisms described above.

The document does not prescribe a strict order that a client must follow for discovery. An implementation may choose to perform all the above steps in parallel for discovery OR choose to follow any desired order and stop the discovery procedure if a mechanism succeeds.

On hosts with more than one interface or address family (IPv4/v6), the TURN server discovery procedure has to be performed for each combination of interface and address family. A client MAY optionally choose to perform the discovery procedure only for a desired interface/address combination if the client does not wish to discover a TURN server for all combinations of interface and address family.

4. Discovery using Service Resolution

This mechanism is performed in two steps:

1. A DNS domain name is retrieved for each combination of interface and address family.

2. Retrieved DNS domain names are then used for S-NAPTR lookups as per [RFC5928]. Further DNS lookups may be necessary to determine TURN server IP address(es).

4.1. Retrieving Domain Name

A client has to determine the domain in which it is located. The following sections provide two possible mechanisms to learn the domain name, but other means of retrieving domain names may be used, which are outside the scope of this document e.g. local configuration.

Implementations may allow the user to specify a default name that is used if no specific name has been configured.

4.1.1. DHCP

DHCP can be used to determine the domain name related to an interface's point of network attachment. Network operators may provide the domain name to be used for service discovery within an access network using DHCP. Sections 3.2 and 3.3 of [RFC5986] define DHCP IPv4 and IPv6 access network domain name options to identify a domain name that is suitable for service discovery within the access network. [RFC2132] defines the DHCP IPv4 domain name option; while this option is less suitable, it may still be useful if the options defined in [RFC5986] are not available.

For IPv6, the TURN server discovery procedure MUST try to retrieve DHCP option 57 (OPTION_V6_ACCESS_DOMAIN). If no such option can be retrieved, the procedure fails for this interface. For IPv4, the TURN server discovery procedure MUST try to retrieve DHCP option 213 (OPTION_V4_ACCESS_DOMAIN). If no such option can be retrieved, the procedure SHOULD try to retrieve option 15 (Domain Name). If neither option can be retrieved the procedure fails for this interface. If a result can be retrieved it will be used as an input for S-NAPTR resolution.

4.1.2. From own Identity

For a TURN client with an understanding of the protocol mechanics of calling applications, the client may wish to extract the domain name from its own identity i.e canonical identifier used to reach the user.

Example

```
SIP      : 'sip:alice@example.com'
Bare JID : 'alice@example.com'
email    : 'alice@example.com'
```

'example.com' is retrieved from the above examples.

The means to extract the domain name may be different based on the type of identifier and is outside the scope of this document.

4.2. Resolution

Once the TURN discovery procedure has retrieved domain names, the resolution mechanism described in [RFC5928] is followed. An S-NAPTR lookup with 'RELAY' application service and the desired protocol tag is made to obtain information necessary to connect to the authoritative TURN server within the given domain.

In the example below, for domain 'example.net', the resolution algorithm will result in IP address, port, and protocol tuples as follows:

```
example.net.
IN NAPTR 100 10 "" RELAY:turn.udp "" example.net.

example.net.
IN NAPTR 100 10 S RELAY:turn.udp "" _turn._udp.example.net.

_turn._udp.example.net.
IN SRV   0 0 3478 a.example.net.

a.example.net.
IN A     192.0.2.1
IN AAAA  2001:db8:8:4::2
```

Order	Protocol	IP address	Port
1	UDP	192.0.2.1	3478
2	UDP	2001:db8:8:4::2	3478

If no TURN-specific S-NAPTR records can be retrieved, the discovery procedure fails for this domain name (and the corresponding interface and IP protocol version). If more domain names are known, the discovery procedure may perform the corresponding S-NAPTR lookups immediately. However, before retrying a lookup that has failed, a client MUST wait a time period that is appropriate for the encountered error (NXDOMAIN, timeout, etc.).

5. DNS Service Discovery

DNS-based Service Discovery (DNS-SD) [RFC6763] and Multicast DNS (mDNS) [RFC6762] provide generic solutions for discovering services available in a local network. DNS-SD/mDNS define a set of naming rules for certain DNS record types that they use for advertising and discovering services.

Section 4.1 of [RFC6763] specifies that a service instance name in DNS-SD has the following structure:

<Instance> . <Service> . <Domain>

The <Domain> portion specifies the DNS sub-domain where the service instance is registered. It may be "local.", indicating the mDNS local domain, or it may be a conventional domain name such as "example.com.". The <Service> portion of the TURN service instance name MUST be "_turn._udp" or "_turn._tcp" or "_turns._udp" or "_turns._tcp".

For example, the following DNS records would be used for a TURN server with instance name "exampleco TURN Server" providing TURN service over UDP on port 5030:

```
_turn._udp.local.
PTR   "exampleco TURN Server"._turn._udp.local.

"exampleco TURN Server"._turn._udp.local.
SRV   0 0 5030 example-turn-server.local.

example-turn-server.local.
A     198.51.100.2

example-turn-server.local.
AAAA  2001:db8:8:4::2
```

5.1. mDNS

A TURN client tries to discover the TURN servers being advertised in the site by multicasting a PTR query "_turn._udp.local." or "_turn._tcp.local" or "_turns._udp.local." or "_turns._tcp.local" or the TURN server can send out gratuitous multicast DNS answer packets whenever it starts up, wakes from sleep, or detects a chance in network configuration. TURN clients receive these gratuitous packet and cache the information contained in it.

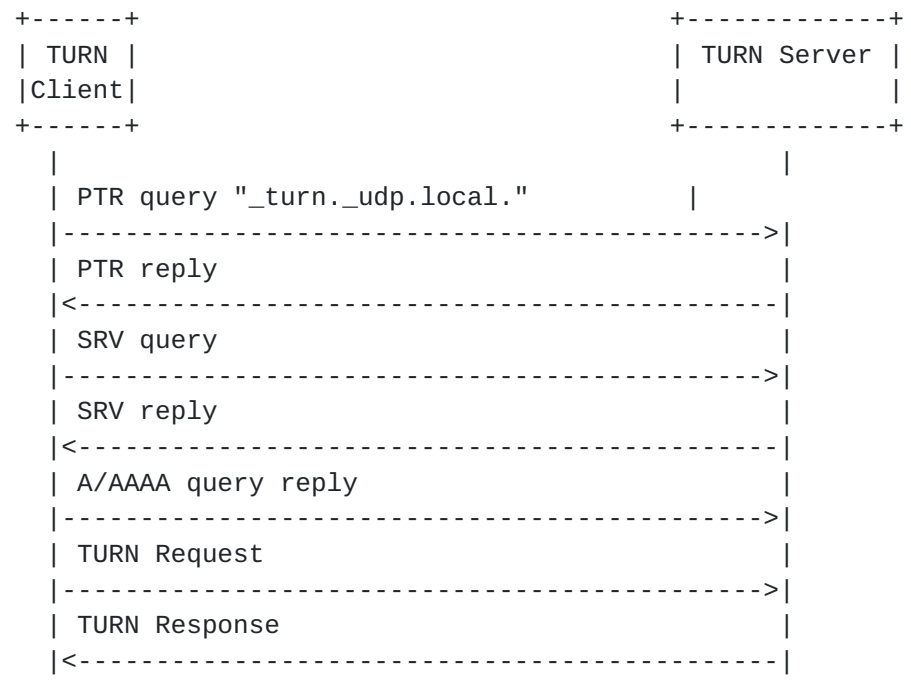

Figure 1: TURN Server Discovery using mDNS

6. Discovery using Anycast

IP anycast can also be used for TURN service discovery. A packet sent to an anycast address is delivered to the "topologically nearest" network interface with the anycast address. Using the TURN anycast address, the only two things that need to be deployed in the network are the two things that actually use TURN.

When a client requires TURN services, it sends a TURN allocate request to the assigned anycast address. The TURN anycast server responds with a 300 (Try Alternate) error as described in [RFC5766]; The response contains the TURN unicast address in the ALTERNATE-SERVER attribute. For subsequent communication with the TURN server, the client uses the responding server's unicast address. This has to be done because two packets addressed to an anycast address may reach two different anycast servers. The client, thus, also needs to ensure that the initial request fits in a single packet. An implementation may choose to send out every new request to the anycast address to learn the closest TURN server each time.

7. Deployment Considerations

7.1. Mobility and Changing IP addresses

A change of IP address on an interface may invalidate the result of the TURN server discovery procedure. For instance, if the IP address assigned to a mobile host changes due to host mobility, it may be required to re-run the TURN server discovery procedure without relying on earlier gained information. New requests should be made to the newly learned TURN servers learned after TURN discovery re-run. However, if an earlier learned TURN server is still accessible using the new IP address, procedures described for mobility using TURN defined in [I-D.ietf-tram-turn-mobility] can be used for ongoing streams.

7.2. Recursively Encapsulated TURN

WebRTC endpoints SHOULD treat any TURN server discovered through the mechanims described in this specification as an enterprise/gateway server, in accordance with Recursively Encapsulated TURN [I-D.ietf-rtcweb-return].

8. IANA Considerations

8.1. Anycast

IANA should allocate an IPv4 and an IPv6 well-known TURN anycast address. 192.0.0.0/24 and 2001:0000::/48 are reserved for IETF Protocol Assignments, as listed at

<http://www.iana.org/assignments/iana-ipv4-special-registry/> and

<http://www.iana.org/assignments/iana-ipv6-special-registry/>

9. Security Considerations

Use of Session Traversal Utilities for NAT (STUN) [RFC5389] authentication is OPTIONAL for TURN servers provided by the local network or by the access network. A network provided TURN server MAY be configured to accept Allocation requests without STUN authentication, and a TURN client MAY be configured to accept Allocation success responses without STUN authentication from a network provided TURN server. In order to protect against man-in-the-middle attacks when accepting a TURN allocation response without STUN authentication, it is RECOMMENDED that the TURN client use one of the following techniques with (D)TLS to validate the TURN server:

- For certificate-based authentication, a pre-populated trust anchor store [RFC6024] allows a TURN client to perform path validation for the server certificate obtained during the (D)TLS handshake.

If the client used a domain name to discover the TURN server, that domain name also provides a mechanism for validation of the TURN server. The client MUST use the rules and guidelines given in section 6 of [RFC6125] to validate the TURN server identity.

- For TURN servers that don't have a certificate trust chain (e.g., because they are on a home network or a corporate network), a configured list of TURN servers can contain the Subject Public Key Info (SPKI) fingerprint of the TURN servers. The public key is used for the same reasons HTTP pinning [RFC7469] uses the public key.

- Raw public key-based authentication, as defined in [RFC7250], could also be used to authenticate a TURN server.

An auto-discovered TURN server is considered to be only as trusted as the path between the client and the TURN server. In order to safely use auto-discovered TURN servers for sessions with 'strict privacy' requirements, the user needs to be able to define privacy criteria (e.g. a trusted list of servers, networks, or domains) that are considered acceptable for such traffic. Any discovered TURN server outside the criteria is considered untrusted and is not used for privacy sensitive communication.

In some auto-discovery scenarios, it might not be possible for the TURN client to use (D)TLS authentication to validate the TURN server. However, fall-back to clear text in such cases could leave the TURN client open to on-path injection of spoofed TURN messages. Instead, a TURN client SHOULD fall-back to encryption-only (D)TLS when (D)TLS authentication is not available. Another reason to fall-back to encryption-only is for privacy, which is analogous to SMTP opportunistic encryption [RFC7435] where one does not require privacy but one desires privacy when possible.

It is suggested that a TURN client attempt (D)TLS with authentication and encryption, falling back to encryption-only if the TURN server cannot be authenticated via (D)TLS. The TURN server could fall back to clear text if it does not support unauthenticated (D)TLS, but fallback to clear text is NOT RECOMMENDED because it makes the client more susceptible to man-in-the-middle attacks and on-path packet injection.

9.1. Service Resolution

The primary attack against the methods described in this document is one that would lead to impersonation of a TURN server. An attacker could attempt to compromise the S-NAPTR resolution. Security considerations described in [RFC5928] are applicable here as well.

In addition to considerations related to S-NAPTR, it is important to recognize that the output of this is entirely dependent on its input. An attacker who can control the domain name can also control the final result. Because more than one method can be used to determine the domain name, a host implementation needs to consider attacks against each of the methods that are used.

If DHCP is used, the integrity of DHCP options is limited by the security of the channel over which they are provided. Physical security and separation of DHCP messages from other packets are commonplace methods that can reduce the possibility of attack within an access network; alternatively, DHCP authentication [RFC3188] can provide a degree of protection against modification. When using DHCP discovery, clients are encouraged to use unicast DHCP INFORM queries instead of broadcast queries which are more easily spoofed in insecure networks.

9.2. DNS Service Discovery

Since DNS-SD is just a specification for how to name and use records in the existing DNS system, it has no specific additional security requirements over and above those that already apply to DNS queries and DNS updates. For DNS queries, DNS Security Extensions (DNSSEC) [RFC4033] should be used where the authenticity of information is important. For DNS updates, secure updates [RFC2136][RFC3007] should generally be used to control which clients have permission to update DNS records.

For mDNS, in addition to what has been described above, a principal security threat is a security threat inherent to IP multicast routing and any application that runs on it. A rogue system can advertise that it is a TURN server. Discovery of such rogue systems as TURN servers, in itself, is not a security threat if there is a means for the TURN client to authenticate and authorize the discovered TURN servers.

9.3. Anycast

In a network without any TURN server that is aware of the TURN anycast address, outgoing TURN requests could leak out onto the external Internet, possibly revealing information.

Using an IANA-assigned well-known TURN anycast address enables border gateways to block such outgoing packets. In the default-free zone, routers should be configured to drop such packets. Such configuration can occur naturally via BGP messages advertising that no route exists to said address.

Sensitive clients that do not wish to leak information about their presence can set an IP TTL on their TURN requests that limits how far they can travel into the public Internet.

10. Acknowledgements

The authors would like to thank Simon Perrault, Paul Kyzivat, Troy Shields, Eduardo Gueiros, Ted Hardie, Bernard Aboba, Karl Stahl, Brian Weis, Ralph Dromes and Brandon Williams for their review and valuable comments. Thanks to Adam Roach for his detailed review and suggesting DNS Service Discovery as an additional discovery mechanism.

11. References

11.1. Normative References

[RFC1035] Mockapetris, P., "Domain names - implementation and specification", STD 13, RFC 1035, DOI 10.17487/RFC1035, November 1987, <http://www.rfc-editor.org/info/rfc1035>.

[RFC2119] Bradner, S., "Key words for use in RFCs to Indicate Requirement Levels", BCP 14, RFC 2119, DOI 10.17487/RFC2119, March 1997, <http://www.rfc-editor.org/info/rfc2119>.

[RFC2132] Alexander, S. and R. Droms, "DHCP Options and BOOTP Vendor Extensions", RFC 2132, DOI 10.17487/RFC2132, March 1997, <http://www.rfc-editor.org/info/rfc2132>.

[RFC2136] Vixie, P., Ed., Thomson, S., Rekhter, Y., and J. Bound, "Dynamic Updates in the Domain Name System (DNS UPDATE)", RFC 2136, DOI 10.17487/RFC2136, April 1997, <http://www.rfc-editor.org/info/rfc2136>.

[RFC3007] Wellington, B., "Secure Domain Name System (DNS) Dynamic Update", RFC 3007, DOI 10.17487/RFC3007, November 2000, <http://www.rfc-editor.org/info/rfc3007>.

[RFC4033] Arends, R., Austein, R., Larson, M., Massey, D., and S. Rose, "DNS Security Introduction and Requirements", RFC 4033, DOI 10.17487/RFC4033, March 2005, <http://www.rfc-editor.org/info/rfc4033>.

[RFC5198] Klensin, J. and M. Padlipsky, "Unicode Format for Network Interchange", RFC 5198, DOI 10.17487/RFC5198, March 2008, <http://www.rfc-editor.org/info/rfc5198>.

[RFC5389] Rosenberg, J., Mahy, R., Matthews, P., and D. Wing, "Session Traversal Utilities for NAT (STUN)", RFC 5389, DOI 10.17487/RFC5389, October 2008, <http://www.rfc-editor.org/info/rfc5389>.

[RFC5766] Mahy, R., Matthews, P., and J. Rosenberg, "Traversal Using Relays around NAT (TURN): Relay Extensions to Session Traversal Utilities for NAT (STUN)", RFC 5766, DOI 10.17487/RFC5766, April 2010, <http://www.rfc-editor.org/info/rfc5766>.

[RFC5928] Petit-Huguenin, M., "Traversal Using Relays around NAT (TURN) Resolution Mechanism", RFC 5928, DOI 10.17487/RFC5928, August 2010, <http://www.rfc-editor.org/info/rfc5928>.

[RFC5986] Thomson, M. and J. Winterbottom, "Discovering the Local Location Information Server (LIS)", RFC 5986, DOI 10.17487/RFC5986, September 2010, <http://www.rfc-editor.org/info/rfc5986>.

[RFC6762] Cheshire, S. and M. Krochmal, "Multicast DNS", RFC 6762, DOI 10.17487/RFC6762, February 2013, <http://www.rfc-editor.org/info/rfc6762>.

[RFC6763] Cheshire, S. and M. Krochmal, "DNS-Based Service Discovery", RFC 6763, DOI 10.17487/RFC6763, February 2013, <http://www.rfc-editor.org/info/rfc6763>.

11.2. Informative References

[I-D.ietf-rtcweb-overview]
Alvestrand, H., "Overview: Real Time Protocols for Browser-based Applications", draft-ietf-rtcweb-overview-15 (work in progress), January 2016.

[I-D.ietf-rtcweb-return]
Schwartz, B. and J. Uberti, "Recursively Encapsulated TURN (RETURN) for Connectivity and Privacy in WebRTC", draft-ietf-rtcweb-return-01 (work in progress), January 2016.

[I-D.ietf-tram-turn-mobility]
Reddy, T., Wing, D., Patil, P., and P. Martinsen, "Mobility with TURN", draft-ietf-tram-turn-mobility-05 (work in progress), August 2016.

[RFC3188] Hakala, J., "Using National Bibliography Numbers as Uniform Resource Names", RFC 3188, DOI 10.17487/RFC3188, October 2001, <http://www.rfc-editor.org/info/rfc3188>.

[RFC5128] Srisuresh, P., Ford, B., and D. Kegel, "State of Peer-to-Peer (P2P) Communication across Network Address Translators (NATs)", RFC 5128, DOI 10.17487/RFC5128, March 2008, <http://www.rfc-editor.org/info/rfc5128>.

[RFC6024] Reddy, R. and C. Wallace, "Trust Anchor Management Requirements", RFC 6024, DOI 10.17487/RFC6024, October 2010, <http://www.rfc-editor.org/info/rfc6024>.

[RFC6125] Saint-Andre, P. and J. Hodges, "Representation and Verification of Domain-Based Application Service Identity within Internet Public Key Infrastructure Using X.509 (PKIX) Certificates in the Context of Transport Layer Security (TLS)", RFC 6125, DOI 10.17487/RFC6125, March 2011, <http://www.rfc-editor.org/info/rfc6125>.

[RFC7250] Wouters, P., Ed., Tschofenig, H., Ed., Gilmore, J., Weiler, S., and T. Kivinen, "Using Raw Public Keys in Transport Layer Security (TLS) and Datagram Transport Layer Security (DTLS)", RFC 7250, DOI 10.17487/RFC7250, June 2014, <http://www.rfc-editor.org/info/rfc7250>.

[RFC7435] Dukhovni, V., "Opportunistic Security: Some Protection Most of the Time", RFC 7435, DOI 10.17487/RFC7435, December 2014, <http://www.rfc-editor.org/info/rfc7435>.

[RFC7469] Evans, C., Palmer, C., and R. Sleevi, "Public Key Pinning Extension for HTTP", RFC 7469, DOI 10.17487/RFC7469, April 2015, <http://www.rfc-editor.org/info/rfc7469>.

Appendix A. Change History

[Note to RFC Editor: Please remove this section prior to publication.]

A.1. Change from draft-patil-tram-serv-disc-00 to -01

- Added IP address (Section 4.1.2) and Own identity (4.1.3) as new means to obtain domain names
- New Section 4.2.1 SOA (inspired by draft-kist-alto-3pdisc)
- 300 (Try Alternate) response for Anycast

A.2. Change from draft-ietf-tram-turn-server-discovery-01 to 02

- Removed sections that describe reverse IP lookup
- Added DNS Service Discovery as an additional discovery mechanism

Authors' Addresses

Prashanth Patil
Cisco Systems, Inc.
Bangalore
India

Email: praspati@cisco.com

Tirumaleswar Reddy
Cisco Systems, Inc.
Cessna Business Park, Varthur Hobli
Sarjapur Marathalli Outer Ring Road
Bangalore, Karnataka 560103
India

Email: tireddy@cisco.com

Dan Wing
Cisco Systems, Inc.
170 West Tasman Drive
San Jose, California 95134
USA

Email: dwing@cisco.com