Workgroup: LSR Working Group
Internet-Draft: draft-li-lsr-droid-00
Published: 4 April 2022
Intended Status: Standards Track
Expires: 6 October 2022
Authors: T. Li
Juniper Networks

Distributed Routing Object Information Database (DROID)

Abstract

Over time, the routing protocols have been burdended with the responsiblity of carrying a variety of information that is not directly relevant to their mission. This includes VPN parameters, configuration information, and capability data. All of the additional data impacts the performance and stability of the routing protocols negatively.

This has been convenient since the backbone of a routing protocol is a small distributed database of routing information. Any service needing a distributed database has considered injecting its data into a routing protocol so that it can leverage the protocols database service. Architecturally, this is a mistake that puts the protocol at risk from undue complexity and overhead.

To avoid this, DROID is a subsystem that is tangential to, but independent of the routing protocols, and provides distributed database services for other routing services. It is based on the publish-subscribe (pub/sub) architecture and is intentionally crafted to be an open mechanism for the transport of ancillary data.

Status of This Memo

Copyright Notice

Table of Contents

1. Introduction

Over time, the routing protocols have been burdended with the responsiblity of carrying a variety of information that is not directly relevant to their mission. This includes VPN parameters, configuration information, and capability data. All of the additional data impacts the performance and stability of the routing protocols negatively.

This has been convenient since the backbone of a routing protocol is a small distributed database of routing information. Any service needing a distributed database has considered injecting its data into a routing protocol so that it can leverage the protocols database service. Architecturally, this is a mistake that puts the protocol at risk from undue complexity and overhead.

To avoid this, DROID is a subsystem that is tangential to, but independent of the routing protocols, and provides distributed database services for other routing services. It is based on the publish-subscribe (pub/sub) architecture and is intentionally crafted to be an open mechanism for the transport of ancillary data.

The service itself runs on OSPF [RFC2328] [RFC5340] Area Border Routers (ABRs) or IS-IS [ISO10589] L1-L2 routers. For brevity, we will use the term 'ABRs' for both cases.

This service uses a simple, hierarchical publish-subscribe architecture. Clients are nodes within non-backbone OSPF areas or L1 IS-IS area. They subscribe with their local ABRs. The ABRs are fully meshed, with the exception that ABRs of the same area need not interact. Notifications initiated by an ABR flow to other ABRs and from there to client nodes.

The availability of this service is advertised as part of the IGP, so that discovery of the service is automatic. Clients can automatically detect their local ABRs and ABRs can detect each other and automatically form the necessary hierarchy.

The protocol runs on top of TCP [RFC0793] and/or QUIC [RFC9000] for reliability. Security is provided by conventional transport protocol mechanisms, such as TLS [RFC5246].

1.1. Use Case: Node Liveness

Overlay services are increasingly common and are implemented by creating tunnels over a physical infrastructure. The failure of one of the tunnel endpoints implies that the traffic towards that endpoint will be lost until the other endpoint recognizes the situation and takes remedial action. Prompt notification of the failure of the other endpoint is useful in minimizing the duration of the outage.

Some network designs have come to rely on examining the IGP's Link State Database (LSDB) to determine node liveness and, through the IGP SPF computation, the node's reachability. However, if the network is to scale, some form of summarization must be employed, resulting in this information no longer being directly available. DROID can address this need by combining its distributed database capabilities with the ability to infer knowledge learned from the IGP.

Node liveness should not be confused with service liveness. If a node is alive, then a service may or may not be up. This protocol only tries to convey node liveness.

1.2. Use Case: Capabilities

Different nodes in the network have different capabilities. Other nodes need to know what these capabilities are for a variety of purposes. The management plane could learn and distribute this information, but asking all nodes to retain all of this information is not efficient. Rather, this information should be made available to the nodes that need the information, when they need it.

Capability information has been carried in the IGP frequently, but when the capabilities are not directly related to the IGP, it is an overuse of the IGP itself. This would be a good application of DROID. Each node should be able to advertise its capabilities into DROID. Interested nodes should be able to request capability information from DROID about any node in the network.

2. Requirements Language

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 BCP 14 [RFC2119] [RFC8174] when, and only when, they appear in all capitals, as shown here.

3. DROID Capability Advertisement

DROID itself is run by ABRs and is advertised in the IGP for connections by clients and other ABRs. Advertisements are done both into the backbone (L2) and into non-backbone (L1) areas. The advertisements into the backbone allow ABRs to automatically mesh.

The advertisements into the non-backbone areas allow clients to automatically determine where the service is available.

3.1. DROID Advertisement in IS-IS

An ABR advertises the IS-IS DROID sub-TLV as part of the IS-IS Router Capability TLV [RFC7981]. This is injected into the ABRs L1 and L2 LSP. The format of the IS-IS Node Liveness sub-TLV is:

```
 0                   1                   2                   3
 0 1 2 3 4 5 6 7 8 9 0 1 2 3 4 5 6 7 8 9 0 1 2 3 4 5 6 7 8 9 0 1
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|     Type      |     Length    |O|N|  Reserved |      TPI      |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|           Port Number         |         IPv4 Address          |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|           IPv4 Address        |         IPv6 Address...       |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
```

Type: TBD1

Length: n * (4 octets + 4 octets if O is set + 16 octets if N is set)

O: 1 if an IPv4 Address is included

N: 1 if an IPv6 Address is included

Reserved: must be zero and ignored on receipt, 6 bits

TPI: Transport Protocol Identifier, 1 octet

0: TCP

1: QUIC

Port Number: Transport protocol port number, 2 octets

IPv4 Address: Service contact address, 4 octets if the O bit is set, 0 otherwise.

IPv6 Address: Service contact address, 16 octets if the N bit is set, 0 otherwise.

The advertisement of this capability indicates that the node is providing the DROID service on the designated port using the designated protocol. The TPI indicates the transport protocol to be used and the Port Number indicates the associated port to be used. The TPI and Port Number pair may be included multiple times to indicate that multiple protocols and port numbers are available. The length of the sub-TLV can be used to determine the number of TPI and Port Number pairs.

An IP address for the ABR MUST be included so that correspondents will know how to access the service. An ABR MUST provide an IPv4 address, an IPv6 address, or both.

3.2. DROID Advertisement in OSPF

The availabilty of the DROID service is provided by the OSPF Node Liveness Sub-TLV. The OSPF Node Liveness Sub-TLV is used by both OSPFv2 and OSPFv3. The semantics are the same as the IS-IS Node Liveness Sub-TLV. The format of the OSPF DROID Sub-TLV is:

```
    0                   1                   2                   3
    0 1 2 3 4 5 6 7 8 9 0 1 2 3 4 5 6 7 8 9 0 1 2 3 4 5 6 7 8 9 0 1
   +-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
   |             Type              |             Length            |
   +-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
   |O|N|  Reserved |      TPI      |           Port Number         |
   +-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
   |                          IPv4 Address                         |
   +-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
   |                          IPv6 Address...                      |
   +-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
```

Type: TBD2

Length: n * 3 octets

O: 1 if an IPv4 Address is included

N: 1 if an IPv6 Address is included

Reserved: must be zero and ignored on receipt, 6 bits

TPI: Transport Protocol Identifier, 1 octet

0: TCP

1: QUIC

Port Number: Transport protocol port number, 2 octets

IPv4 Address: Service contact address, 4 octets if the O bit is set, 0 otherwise.

IPv6 Address: Service contact address, 16 octets if the N bit is set, 0 otherwise.

The TPI and Port Number fields are used in the same way as for IS-IS.

4. DROID

4.1. Messages

DROID sends messages in a stream inside of the selected transport protocol. The protocol uses three message types:

Publish: A node generates a Publish message to change a data value in the database. If another node has subscribed to this data item, it will be informed by a Notification message.

Subscribe: A Subscribe message creates a subscription for a set of data items. Subsequent updates for the data will generate a corresponding Notification message containing the data items.

Notification: A Notification message is generated when a database item is modified. Any nodes that have subscribed to the data item are sent a Notification message with the value of the data item.

Each message has sub-TLVs to carry more specific information.

4.2. Keys

Each item in the database must have a key. The key space is hierarchical and variable length. Traditionally, keys have been an ASCII string, with levels in the hierarchy separated by the '/' character, but this is extremely ineffcient. A hierarchical binary key would be more efficient but is harder to manage.

Definition of the key space is out of scope for this document.

4.3. Object Values

An object in the database is an opaque, variable length string of octets. The interpretation of an object value is outside of the scope of this document.

4.4. Client Actions

The client may determine the set of ABRs that it wishes to communicate with by examination of its LSDB. The client SHOULD open connections to at least two ABRs for redundancy. If the client cannot open two connections, then the management system should be informed.

Clients send Subscribe messages to subscribe to particular data that it would like to receive Notifications about. A client MAY set the G bit in the Subscribe message if it would like to get the current value of the data as of when it subscribes.

Clients never send Notification messages and never receive Subscribe messages. The actions of the client on receiving a Notification message are out of scope for this document.

4.4.1. Client Liveness Actions

The client MAY send Subscribe messages (with a Liveness Subscribe sub-TLV) on each of its ABR connections. A client MAY subscribe for any number of prefixes, but it is expected that a client will send a subscription for each of the tunnel endpoints that it will correspond with. A client may subscribe for a host (a /32 or /128 prefix) or a shorter prefix.

4.4.2. Client Capability Actions

A client MAY send Publish messages to advertise its own capabilities. A client MAY send Subscribe messages to subscribe for capabilities of other nodes.

There are no special mechanisms to support client capabilities. This is simply a straightforward example of DROID mechanisms.

4.5. ABR Actions

Each ABR MUST advertise the availability of the Node Liveness service into the backbone (L2) area and into any non-backbone (L1) areas.

Each ABR MUST have a single connection to each other ABR that is part of a different non-backbone (L1) area. To prevent duplicate connections, only one ABR should initiate the connection. For IS-IS, the node with the lowest system ID should initiate the connection. For OSPFv4, the node with the lowest IPv4 router ID should initiate the connection. For OSPFv3, the node with the lowest IPv6 router ID should initiate the connection.

Each ABR may receive Subscribe messages, each containing a prefix. These are retained in a Subscription Database (SDB) along with its associated connection information. If a transport connection closes, then all subscriptions associated with the connection should be removed from the SDB. If an ABR receives a Subscription message requesting a prefix be unsubscribed, then the prefix should be removed from the SDB for that connection.

If an ABR receives a Subscribe message for a prefix that is being injected by a non-attached area, then it SHOULD determine the set of ABRs that are advertising that prefix or less specifics and subscribe with only those ABRs. The ABR MAY subscribe for the prefix or any of the less specifics. It is RECOMMENDED that the ABR subscribe for the most specific prefix that is less specific than the original prefix. If the ABR cannot find a matching prefix or less specific prefix, then the ABR MAY subscribe for all of prefixes that are more specific. Extreme caution should be used before subscribing for 0/0.

If the ABR has subscribed for a prefix and that prefix is no longer advertised by another ABR then an ABR MAY unsubscribe, re-evaluate its subscription and subscribe for a different prefix. In this way, if a summary prefix changes, the ABR can shift to the new summary prefix.

An ABR or client SHOULD NOT send duplicate subscriptions. If an ABR or client is already subscribed for a prefix, a duplicate subscription MUST NOT create a duplicate entry in the SDB.

A client may be co-located with an ABR. In other words, an ABR may create subscriptions for its own purposes.

4.5.1. ABR Liveness Actions

Each ABR should monitor its IGP LSDB for changes in node liveness. If an ABR sees an addition to the LSDB, then it is considered an Up Event for that node. If an ABR sees a LSP/LSA time out or become unreachable, then it is considered a Down Event for that node. Up Events and Down Events for non-host prefixes are out of scope for this document.

If an ABR receives a Notification message with an Up Event for a prefix, then it is considered an Up Event for the prefix. If an ABR receives a Notification message with a Down Event for a prefix, then it is considered a Down Event for the prefix.

If an ABR observes an Up Event for a host, it examines its SDB for subscriptions for that node or for any less specific prefixes. If there are any, then the ABR sends a Notification message (with a Liveness Notification sub-TLV) with an Up Event for that host to each node that subscribed. If there are no subscriptions, then the event MUST be ignored.

Similarly, if an ABR observes a Down Event for a host, it examines its SDB for subscriptions for that node or for any less specific prefixes. If there are any, then the ABR sends a Notification message (with a Liveness Notification sub-TLV) with a Down Event for that host to each node that subscribed. If there are no subscriptions, then the event MUST be ignored.

4.5.2. Autonomous Notification Mode

This section describes OPTIONAL ABR behavior.

An ABR MAY learn a set of prefixes from its management plane and enter those prefixes into its SDB. Upon an Up or Down Event for such a prefix, the ABR MAY send corresponding notification messages to all other ABRs.

This may cause ABRs to receive unexpected Notification messages. Since these do not match client subscription messages in its own SDB, such messages SHALL be ignored.

4.5.3. Proxy ABRs

Another node may perform ABR functions instead of the ABR itself. The alternate node is a 'proxy ABR' and performs all of the functions of the ABR with respect to this protocol, except for injecting capability advertisements into the LSDB. The proxy ABR should listen to the IGP within the area so that it can correctly generate notifications. The proxy ABR must also listen to the backbone or L2 area so that it can locate other ABRs. One or more ABRs SHOULD advertise the availability of the proxy ABR in its capability advertisements. How the real ABRs learn about the proxy ABR is out of scope for this document.

4.6. Publish Messages

A Publish message has the following format:

```
    0                   1                   2                   3
    0 1 2 3 4 5 6 7 8 9 0 1 2 3 4 5 6 7 8 9 0 1 2 3 4 5 6 7 8 9 0 1
   +-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
   |     Type      |            Length             |    Sub-TLVs ...
   +-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
```

Type: 1 (Publish message), 1 octet

Length: length of the sub-TLVs, 2 octets

Sub-TLVs: One or more sub-TLVs, specifying the subscription/ unsubscription. Variable length.

4.7. Subscribe Messages

A Subscribe message has the following format:

```
 0                   1                   2                   3
 0 1 2 3 4 5 6 7 8 9 0 1 2 3 4 5 6 7 8 9 0 1 2 3 4 5 6 7 8 9 0 1
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|     Type      |           Length              |S|G| Reserved  |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
| Sub-TLVs ...
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
```

Type: 2 (Subscribe message), 1 octet

Length: 1 + length of the sub-TLVs, 2 octets

S: 1 bit

0: Subscribe

1: Unsubscribe

G: if set, then the receiver should generate an immediate Notification with the data value(s), 1 bit

Reserved: must be zero and ignored on receipt, 6 bits

Sub-TLVs: One or more sub-TLVs, specifying the subscription/ unsubscription. Variable length.

Use of the G bit for large queries can generate large amounts of data.

4.8. Notification Messages

A Notification message has the following format:

```
 0                   1                   2                   3
 0 1 2 3 4 5 6 7 8 9 0 1 2 3 4 5 6 7 8 9 0 1 2 3 4 5 6 7 8 9 0 1
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|     Type      |          Length               |  Sub-TLVs ...
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
```

Type: 3 (Notification message), 1 octet

Length: length of the sub-TLVs, 2 octets

Sub-TLVs: One or more sub-TLVs, specifying the subscription and data value(s). Variable length.

4.9. Message Sub-TLVs

The following sub-TLVs may be used with any of the messages above. Multiple sub-TLVs are expected to be used in combination to qualify the containing message. Type codes for DROID Sub-TLVs are allocated from the "DROID Sub-TLV Types" registry, defined below.

4.9.1. Prefix sub-TLV

```
    0                   1                   2                   3
    0 1 2 3 4 5 6 7 8 9 0 1 2 3 4 5 6 7 8 9 0 1 2 3 4 5 6 7 8 9 0 1
   +-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
   |     Type      |           Length              |  Prefix len   |
   +-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
   |              AFI              |    Prefix ...
   +-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
```

Type: 1, 1 octet

Length: 3 + the number of octets for the prefix, 2 octets

AFI: Address Family Identifier [afireg], 2 octets

Prefix len: number of significant bits in the prefix, 1 octet

Prefix: n octets

4.9.2. Key Sub-TLV

The Key sub-TLV has the format:

```
    0                   1                   2                   3
    0 1 2 3 4 5 6 7 8 9 0 1 2 3 4 5 6 7 8 9 0 1 2 3 4 5 6 7 8 9 0 1
   +-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
   |     Type      |         Length                | Key ....
   +-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
```

Type: 2, 1 octet

Length: length of the Key field, in octets, 2 octets

Key: variable length

The Key is an opaque variable length list of octets.

4.9.3. Object Value Sub-TLV

The Object Value sub-TLV has the format:

```
    0                   1                   2                   3
    0 1 2 3 4 5 6 7 8 9 0 1 2 3 4 5 6 7 8 9 0 1 2 3 4 5 6 7 8 9 0 1
   +-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
   |     Type      |         Length                | Object Value ...
   +-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
```

Type: 3, 1 octet

Length: length of the Object Value field, in octets, 2 octets

Object Value: variable length

The Object Value is an opaque variable length list of octets.

The Object Value sub-TLV should never appear in a Subscribe message.

4.9.4. Liveness Sub-TLV

The Liveness sub-TLV has the format:

```
    0                   1                   2                   3
    0 1 2 3 4 5 6 7 8 9 0 1 2 3 4 5 6 7 8 9 0 1 2 3 4 5 6 7 8 9 0 1
   +-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
   |     Type      |             Length            |U|D| Reserved  |
   +-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
```

Type: 128, 1 octet

Length: 1, 2 octets

U: Up event, 1 bit

D: Down event, 1 bit

Reserved: must be zero and ignored on receipt, 6 bits

Up events and Down events MAY be subscribed independently or jointly.

5. IANA Considerations

5.1. IS-IS

This document requests the following code points from the "IS-IS Sub-TLVs for IS-IS Router CAPABILITY TLV" registry.

Type: TBD 1

Description: IS-IS Node Liveness sub-TLV

Reference: This document

5.2. OSPF

This document requests the following code points from the "OSPF Router Information (RI) TLVs" registry:

Type: TBD 2

Description: OSPF Node Liveness Sub-TLV

Reference: This document

5.3. DROID Parameters

This document requests that IANA create a new Protocol Registry for "DROID Parameters". The initial contents are the "DROID Sub-TLV Types Registry" and the "DROID Capability Values Registry" defined below.

5.4. DROID Sub-TLV Types Registry

This document requests that IANA create a new registry called the "DROID Sub-TLV Types" registry under the "DROID Parameters" protocol registry. For this registry, the registration procedure is "Standards Action". The range of available numeric values is 0-255. Generic sub-TLVs should be allocated from the range of 0-127. Data specific sub-TLVs should be allocated from the range 128-255. The fields in this registry are a "Value" and a "Name". The initial contents of this registry should be:

Value	Name
1	Prefix sub-TLV
2	Key sub-TLV
3	Object Value sub-TLV
128	Liveness sub-TLV

Table 1

5.5. DROID Capability Values Registry

This document requests that IANA create a new registry called the "DROID Capability Values" registry under the "DROID Parameters" protocol registry. For this registry, the registration procedure is "Standards Action". The range of available numeric values is 0-255. There are no initial contents. The fields in this registry are a "Value" and a "Name".

Values in this registry should be allocated in increasing order, starting with zero.

Each value in this registry corresponds to a bit position within the Capabilities field of the Capabilities sub-TLV. Value 0 indicates the most significant bit of the first octet, with subsequent values indicating bits of decreasing signficance and then subsequent octets, starting with the most significant bit. Thus, value 8 would correspond to the most signficant bit of the second octet.

6. Security Considerations

Security of transport protocol connections are addressed by the use of conventional transport protocol security techniques, such as TLS. IGP advertisements are not expected to have privacy, so the advertisement of the service is not a security issue.

Authentication is an outstanding issue, to be handled in a future version of this document.

7. Normative References

[afireg] IANA, "Address Family Numbers", November 1988, <https://www.iana.org/assignments/address-family-numbers/address-family-numbers.xhtml#address-family-numbers-2>.

[ISO10589] ISO, "Intermediate system to Intermediate system routing information exchange protocol for use in conjunction with the Protocol for providing the Connectionless-mode Network Service (ISO 8473)", August 1987, <ISO/IEC 10589:2002>.

[RFC0793] Postel, J., "Transmission Control Protocol", STD 7, RFC 793, DOI 10.17487/RFC0793, September 1981, <https://www.rfc-editor.org/info/rfc793>.

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

[RFC2328] Moy, J., "OSPF Version 2", STD 54, RFC 2328, DOI 10.17487/RFC2328, April 1998, <https://www.rfc-editor.org/info/rfc2328>.

[RFC5246] Dierks, T. and E. Rescorla, "The Transport Layer Security (TLS) Protocol Version 1.2", RFC 5246, DOI 10.17487/

RFC5246, August 2008, <https://www.rfc-editor.org/info/rfc5246>.

[RFC5340] Coltun, R., Ferguson, D., Moy, J., and A. Lindem, "OSPF for IPv6", RFC 5340, DOI 10.17487/RFC5340, July 2008, <https://www.rfc-editor.org/info/rfc5340>.

[RFC7981] Ginsberg, L., Previdi, S., and M. Chen, "IS-IS Extensions for Advertising Router Information", RFC 7981, DOI 10.17487/RFC7981, October 2016, <https://www.rfc-editor.org/info/rfc7981>.

[RFC8174] Leiba, B., "Ambiguity of Uppercase vs Lowercase in RFC 2119 Key Words", BCP 14, RFC 8174, DOI 10.17487/RFC8174, May 2017, <https://www.rfc-editor.org/info/rfc8174>.

[RFC9000] Iyengar, J., Ed. and M. Thomson, Ed., "QUIC: A UDP-Based Multiplexed and Secure Transport", RFC 9000, DOI 10.17487/RFC9000, May 2021, <https://www.rfc-editor.org/info/rfc9000>.

Author's Address

Tony Li
Juniper Networks
1133 Innovation Way
Sunnyvale, California 94089
United States of America

Email: tony.li@tony.li