Network Working Group
Internet Draft
Expiration Date: March 1997

P Doolan
cisco Systems

B Davie
cisco Systems

D Katz
cisco Systems

Y Rekhter
cisco Systems

E Rosen
cisco Systems

September 1996

Tag Distribution Protocol

draft-doolan-tdp-spec-00.txt

Status of this Memo

1. Abstract

An overview of a tag switching architecture is provided in [Rekhter]. This document defines the Tag Distribution Protocol (TDP) referred to in [Rekhter].

TDP is a two party protocol that runs over a connection oriented transport layer with guaranteed sequential delivery. Tag Switching Routers use TDP to communicate tag binding information to their peers. TDP supports multiple network layer protocols including but not limited to IPv4, IPv6, IPX and AppleTalk.

We define here the PDUs and operational procedures for this TDP and specify its transport requirements.

Contents

2. Protocol Overview

A tag switching architecture is described in [Rekhter]. As explained in that document Tag Switching Routers (TSRs) create tag bindings, and then distribute the tag binding information among other TSRs.

TDP provides the means for TSRs to distribute, request, and release tag binding information for multiple network layer protocols. TDP also provides means to open, monitor and close TDP sessions and to indicate errors that occur during those sessions.

TDP is a two party protocol that runs over a connection oriented transport layer with guaranteed sequential delivery.

A TSR that wishes to exchange tag bindings with another opens a transport connection to that other TSR. TSRs identify their remote tag distribution peers using the Router ID of the remote TSR. The means by which a TSR obtains this information is outside the scope of this document. Configuration is one example.

Once a transport connection has been opened then the TSRs exchange TDP PDUs that encode tag binding information. TDP is symmetrical in that once the transport connection has been opened the peer TSRs may each send and receive TDP PDUs at will. A single TSR may have TDP sessions with multiple other TSRs. Each of these sessions is completely independent of the others.

TDP does not require any keepalive notification from the transport, but implements its own keepalive timer. The usage is straightforward: peers must communicate within the period specified by the timer. Each time a TDP peer receives a TDP PDU it resets the timer. If the timer expires some number of times without reception of a TDP PDU from the remote system the TDP closes the session with its peer.

When a TSR determines that it lost a TDP session with another TSR, if the TSR has any tag bindings that were created as a result of receiving tag binding requests from the peer, the TSR may destroy these bindings (and deallocate tags associated with these binding).

When a TSR determines that it lost a TDP session with another TSR, the TSR shall no longer use the binding information it received from the other TSR.

The procedures that govern when other components in a TSR invoke services from TDP and how a TSR maintains its TIBs are beyond the scope of this document.

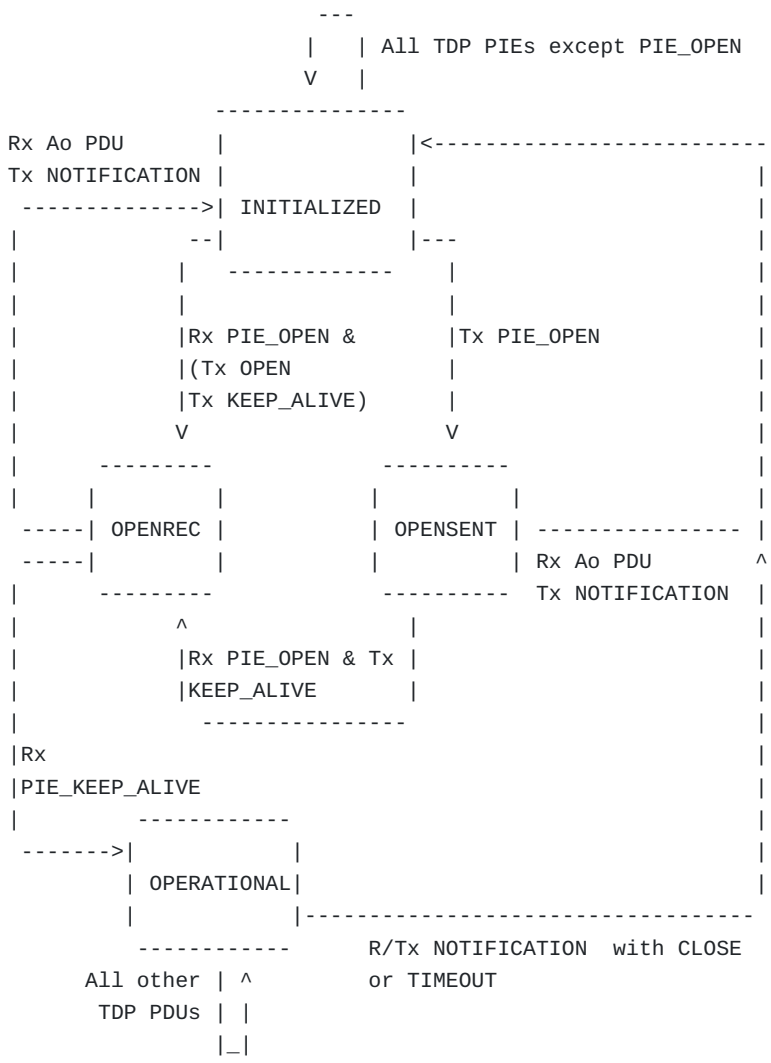
The use of TDP does not preclude the use of other mechanisms to

distribute tag binding information.

3. State machines

It is convenient to describe the TDP's behavior in terms of state machines. We define the TDP state machine to have four possible states and present the behavior as a state transition table and as a state transition diagram.

STATE	EVENT		NEW STATE
	Initialization		INITIALIZED
INITIALIZED	Sent	TDP_PIE_OPEN	OPENSENT
	Received	TDP_PIE_OPEN	OPENREC
OPENREC	Received	TDP_PIE_KEEP_ALIVE	OPERATIONAL
	Received	Any other TDP PDU	INITIALIZED
OPENSENT	Received	TDP_PIE_OPEN &	
	Transmit	TDP_PIE_KEEP_ALIVE	OPENREC
	Received	Any other TDP PDU	INITIALIZED
	Sent	TDP_PIE_NOTIFICATION	INITIALIZED
OPERATIONAL	Rx/Tx	TDP_PIE_NOTIFICATION with CLOSING parameter	INITIALIZED
	Other	TDP PDUs	OPERATIONAL
	Timeout		INITIALIZED



3.1. Transport connections

A TSR that implements TDP opens a transport connection to a peer TSR. Once open, and regardless of which TSR opened it, the transport connection is used bidirectionally. That is there is only one 'connection' used for a TDP session between two TSRs.

3.2. Timeout

Timeout in the state transition table and diagram indicates that the keep alive timer set to HOLD_TIME has expired. See TDP_PIE_OPEN for a discussion of this mechanism.

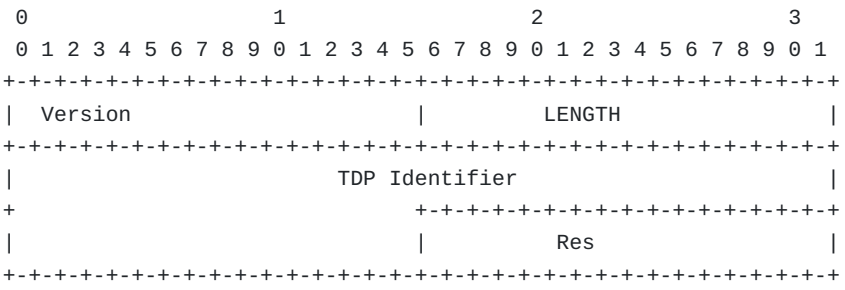
4. Protocol Data Units (PDUs)

TDP PDUs are variable length and consist of a fixed header and one or more Protocol Information Elements (PIE) each with a Type Length Value (TLV) structure. Within a single PIE TLVs may be nested to an arbitrary depth.

A single TDP PDU may contain multiple PIEs. The maximum TDP PDU size is 4096 octets.

4.1. TDP Fixed Header

The fixed header of the TDP PDU is:



Version:

This two octet unsigned integer contains the version number of the protocol. A TDP version number must lie in the range 0x01 < Version < 0xFF. This version of the TDP specification specifies protocol Version = 1.

LENGTH:

This two octet integer specifies the length in octets of the data portion of the PDU. LENGTH is set to the length of the PDU in octets minus four.

TDP Identifier:

Six octet unsigned integer containing a unique identifier for the TSR that generated the PDU. The value of this Identifier is determined on startup. The same value is used no matter which interface(s) of the TSR TDP is running on.

Res:

This field is reserved. It must be set to zero on transmission and must be ignored on receipt.

4.2. TDP TLVs

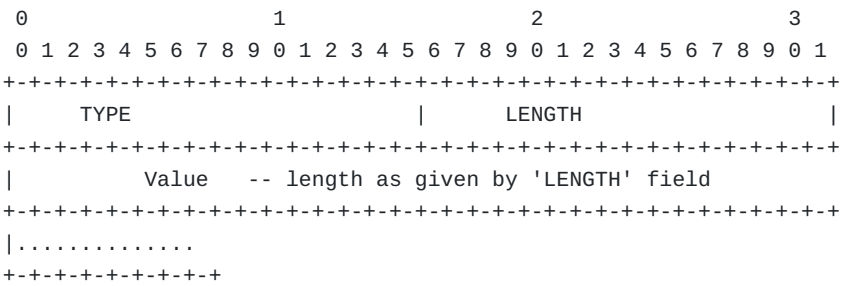
The TDP fixed header frames Protocol Information Elements (PIEs) that have a Type Length Value (TLV) structure.

In this protocol TYPE is a 16 bit integer value that encodes how the VALUE field is to be interpreted. Within a single PIE TLVs may be nested to an arbitrary depth. A TDP must silently discard TLVs that it does not recognize.

LENGTH is an unsigned 16 bit integer value that encodes the length of the VALUE field in octets. LENGTH is set to the length of the whole TLV in octets minus four. A LENGTH of zero indicates that there is no value field present.

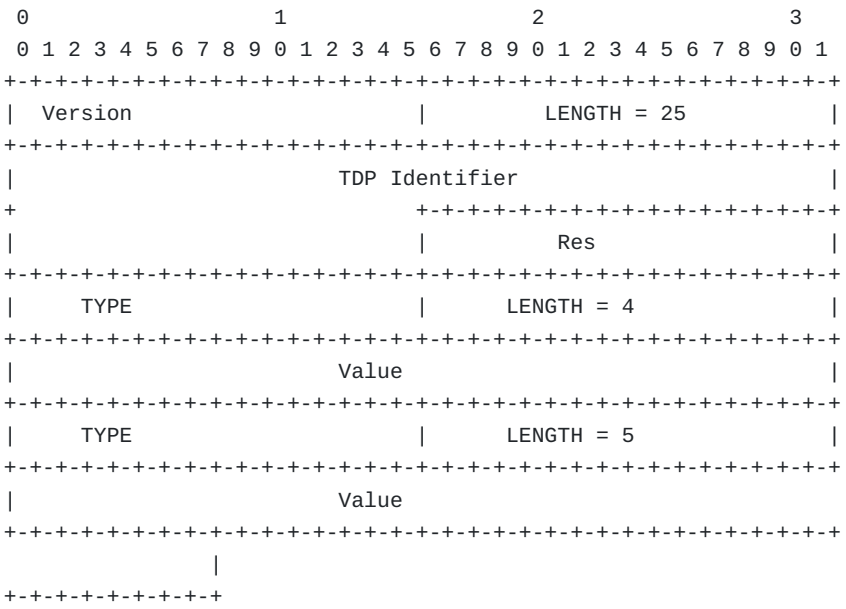
VALUE is an octet string of length LENGTH octets that encodes information the semantics of which are indicated by the TYPE field.

A single TLV has the following format:



4.3. Example TDP PDU

A complete TDP PDU containing two PIEs having 4 and 5 octets of Value field respectively would have the following structure:



4.4. PIEs defined in V1 of TDP

The following PIEs are defined for this version of the protocol. They are described in the sections that follow

```
Type 0x100 TDP_PIE_OPEN
Type 0x200 TDP_PIE_BIND
Type 0x300 TDP_PIE_REQUEST_BIND
Type 0x400 TDP_PIE_REMOVE_BIND
Type 0x500 TDP_PIE_KEEP_ALIVE
Type 0x600 TDP_PIE_NOTIFICATION
Type 0x700 Unassigned
..........
Type 0xFF00
```

Each of these PIEs may have optional TLV encoded parameters as described below.

4.5. TDP_PIE_OPEN

TDP_PIE_OPEN is the first PIE sent by a TSR initiating a TDP session to its peer. It is sent immediately after the transport connection has been opened. The TSR receiving a TDP_PIE_OPEN responds either with a TDP_PIE_KEEPALIVE or with a TDP_PIE_NOTIFICATION.

4.5.1. Initiating a TDP session

A TSR initiating a TDP session sets the TDP_OPEN_PIE's fields as described below, issues a PDU containing it to the target peer, the TDP state machine transitions to the OPENSENT state.

While in the OPENSENT state a TSR takes the following actions:

If it receives an 'acceptable' TDP_PIE_OPEN then then TSR sends a TDP_PIE_KEEPALIVE and the TDP state machine transitions to the OPEN_REC state.

Receipt of any other PDU is an error and results in sending a TDP_PIE_NOTIFICATION indicating a bad open and transition to the INITIALIZED state.

4.5.2. Passive OPEN

A TSR in the INITIALIZED state that receives a TDP_PIE_OPEN behaves as follows:

If it can support the version of the protocol proposed by the TSR that issued the TDP_PIE_OPEN then it sets Version in all its subsequent communication with that TSR to the value proposed in Prop Ver and obeys the rules specified for that version of the protocol.

TSR sends a PDU containing a TDP_PIE_OPEN PIE to the TSR that initiated the TDP session.

TSR sends a PDU containing a TDP_PIE_KEEPALIVE PIE to the TSR that initiated the TDP session.

The TDP state machine transitions to the OPEN_REC state

If the TSR cannot support the version of the protocol proposed in the TDP_PIE_OPEN then it sends a TDP_PIE_NOTIFICATION PDU that informs the TSR which generated the PIE_OPEN of the version(s) it can support. The TDP state machine transitions to the INITIALIZED state. See below under errors for more details.

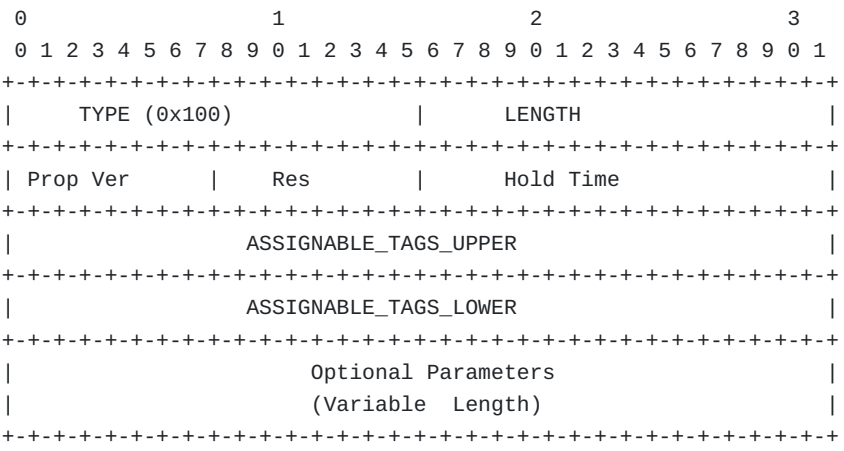
4.5.3. OPENREC state

When in the OPENREC state a TSR takes the following actions:

If a TDP_PIE_KEEPALIVE is received then it transitions to the OPERATIONAL state.

Receipt of any other PDU causes the generation of a TDP_PIE_NOTIFICATION and transition to the INITIALIZED state.

The TDP_PIE_OPEN has the following format



TYPE:

Type field as described above. Set to 0x100 for TDP_PIE_OPEN.

LENGTH:

Length in octets of the value field of this PIE. LENGTH is set to the length of the whole PIE in octets minus four.

Prop Ver:

The Version of the TDP that the TSR that generated this PDU proposes be used for this TDP session once it is established. Note that the session is not established until the TSR that issues a

TDP_PIE_OPEN receives a TDP_PIE_OPEN in response.

Res:

This field is reserved. it must be set to zero on transmission and must be ignored on receipt

ASSIGNABLE_TAGS_UPPER:

This four octet integer represents the upper bound on the value of a tag that the TSR receiving this PDU may insert in an Upstream Tag Allocation. See the description of BLIST_TYPE 1 in the section on TDP_PIE_BIND for more details.

ASSIGNABLE_TAGS_LOWER:

This four octet integer represents the lower bound on the value of a tag that the TSR receiving this PDU may insert in an Upstream Tag Allocation. See the description of BLIST_TYPE 1 in the section on TDP_PIE_BIND for more details.

A TSR may set the value of this field to be higher than the value in the ASSIGNABLE_TAGS_UPPER field. This is used to indicate that the TSR will not support Upstream tag allocation for this TDP session.

Hold Time:

Two octet unsigned non zero integer that indicates the number of seconds that the peer initiating the connection proposes for the value of the Hold Timer. Upon receipt of a PDU with PIE TDP_PIE_OPEN , a TDP peer MUST calculate the value of the Hold Timer by using the smaller of its configured HOLD_TIME and the HOLD_TIME received in the PDU. The value chosen for HOLD_TIME indicates the maximum number of seconds that may elapse between the receipt of successive PDUs from the TDP peer. The Hold Timer is reset each time a TDP_PDU arrives. If the timer expires without the arrival of a TDP_PDU then a TDP_NOTIFICATION with the optional parameter CLOSING is sent.

Optional Parameters:

This variable length field contains zero or more optional PIEs supplied in TLV structures.

OPTIONAL PARAMETER	Type	Length	Value
DOWNSTREAM_ON_DEMAND	0x101	0	0

DOWNSTREAM_ON_DEMAND:

A TSR may supply this optional parameter to indicate that it wishes to use downstream tag allocation on demand. A TSR receiving a TDP_PIE_OPEN containing this optional parameter is required to use TDP_PIE_BIND_REQUEST to obtain bindings from the TSR that issued the TDP_PIE_OPEN.

4.5.4. Errors

All Errors generated by the receipt of a TDP_PIE_OPEN are reported by issuing a TDP_PIE_NOTIFICATION. The value field of the PIE contains one or more TLVs describing individual errors with more precision.

4.5.4.1. TDP_OPEN_UNSUPPORTED_VER:

This error is issued to indicate to the TSR that generated the TDP_PIE_OPEN that this TSR does not support the version of TDP proposed in 'Prop Ver' in the PIE_OPEN. TDP_OPEN_UNSUPPORTED_VER reports the version(s) of the protocol that this TSR does support.

TYPE:

TDP_OPEN_UNSUPPORTED_VER = 0x101

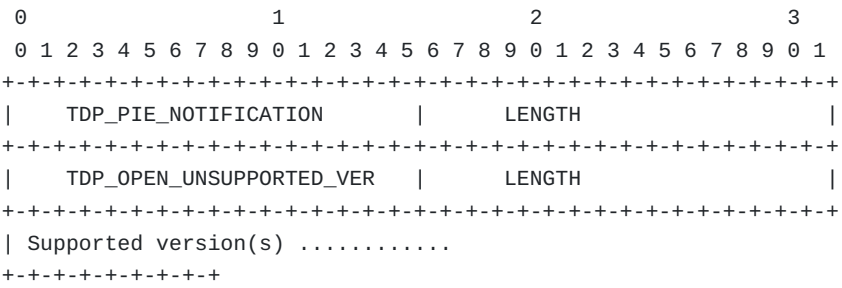
LENGTH:

Length in octets of the value field of this PIE. LENGTH is set to the length of the whole PIE in octets minus four.

VALUE:

One or more 2 octet integers that encode the Version(s) of the protocol that this TSR supports.

The format of an NOTIFICATION PIE containing TDP_OPEN_UNSUPPORTED_VER is:



4.5.4.2. TDP_BAD_OPEN

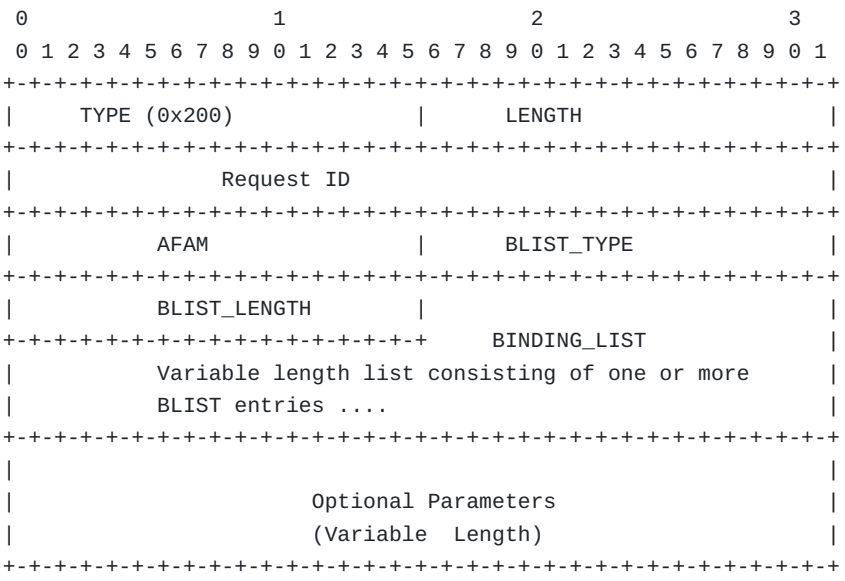
This error is issued to indicate failure during the open phase.

Error	Type	Length	Value
TDP_BAD_OPEN	0x102	0	0

4.6. TDP_PIE_BIND

TDP_PIE_BIND is sent from one TSR to another to distribute tag bindings. Transmission of a TDP_PIE_BIND may occur as a result of some local decision or it may be in response to the reception of a TDP_REQUEST_BIND.

This PIE has the following format



TYPE:

Type field as described above. Set to 0x200 for TDP_PIE_BIND.

LENGTH:

Length in octets of the value field of this PIE. LENGTH is set to the length of the whole PIE in octets minus four.

Request ID:

If this TDP_PIE_BIND is generated in response to a TDP_PIE_REQUEST_BIND then TSR places the value of the Request ID from that request PIE in this field. For all other TDP_PIE_BINDS this field must be set to zero.

AFAM:

This 16 bit integer contains a value from ADDRESS FAMILY NUMBERS in Assigned Numbers [Reynolds] that encodes the address family that the network layer address in the tag bindings in the BINDING_LIST is from. This protocol provides support for multiple network address families.

BLIST_TYPE:

This 16 bit integer contains a value from the table below that encodes the format and semantics of the BLIST entries in the BINDING_LIST field.

BLIST_TYPE	BLIST entry format
0	Null list (see TDP_PIE_REMOVE_BIND)
1	32 bit Upstream assigned
2	32 bit Downstream assigned
3	32 bit Multicast Upstream assigned (*,G)
4	32 bit Multicast Upstream assigned (S,G)

The formats are defined below.

BLIST_LENGTH:

Two octet unsigned integer that encodes the length of the BINDING_LIST

BINDING_LIST:

Variable length field consisting of one or more BLIST entries of the type indicated by BLIST_TYPE.

Optional Parameters:

This variable length field contains zero or more optional PIEs supplied in TLV structures.

4.6.1. BLIST_TYPE 0

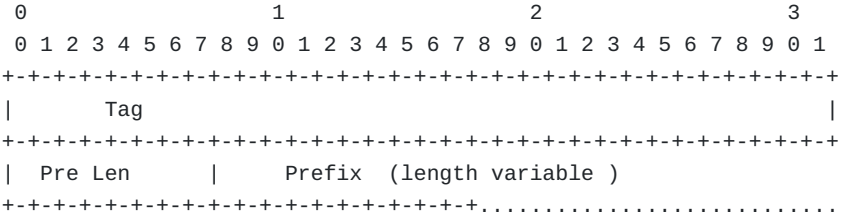
BLIST_TYPE = 0 indicates that there are no BLIST entries. See TDP_PIE_REMOVE_BIND for further details.

4.6.2. BLIST_TYPE 1 and 2

A BLIST_TYPE 1 contains Upstream assigned tags. A TDP must only include tag values in a BLIST_TYPE 1 tag entry that lie between the values, inclusive of those values, that the TSR to whom the TDP_PIE_BIND is being sent indicated it could support during the OPEN phase. A TSR indicates the range of values it can support for Upstream allocation by using the ASSIGNABLE_TAGS_ fields in TDP_PIE_OPEN.

Note that a TSR that indicating ASSIGNABLE_TAGS_LOWER greater than ASSIGNABLE_TAGS_UPPER in the TDP_PIE_OPEN is indicating that it will not allow Upstream tag allocation.

BLIST entries of type 1 and 2 have the following format.



Tag:

32 bit tag value.

Pre Len:

This one octet unsigned integer contains the length in bits of the address prefix that follows.

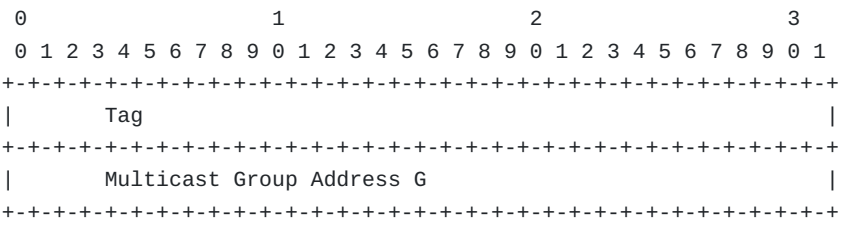
Prefix:

A variable length field containing an address prefix whose length, in bits, was specified in the previous (Pre Len) field. A Prefix is padded with sufficient trailing zero bits to cause the end of the field to fall on an octet boundary.

4.6.3. BLIST_TYPE 3

This binding allows the association of a tag with the (*,G) shared tree. See [Deering] for a discussion of (*,G) shared trees.

The (*,G) binding has the following format:



Tag:

32 bit tag value.

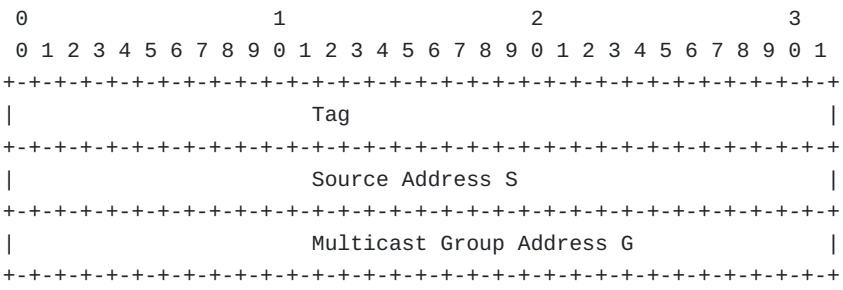
Multicast Group Address G:

Multicast Group Address. The length of this address is network layer specific and can be deduced from the value of AFAM. The diagram above illustrates a four octet IPv4 address format.

4.6.4. BLIST_TYPE 4

This binding type allows association of a tag with a (S,G) source rooted tree. See [Deering] for a discussion of (S,G) trees.

The (S,G) binding has the following format:



Tag:

32 bit tag value.

Source Address S:

Network Layer address of the source sending to the G tree. The length of this address is network layer specific and can be deduced from the value of AFAM. The diagram above illustrates a four octet IPv4 address format.

Multicast Group Address G:

Network Layer Multicast group address. The length of this address is network layer specific and can be deduced from the value of AFAM. The diagram above illustrates a four octet IPv4 address format.

4.7. TDP_PIE_REQUEST_BIND

TDP_PIE_REQUEST_BIND is sent from a TSR to a peer to request a binding for one or more specific NLRIs, or to request all the bindings that its peer has.

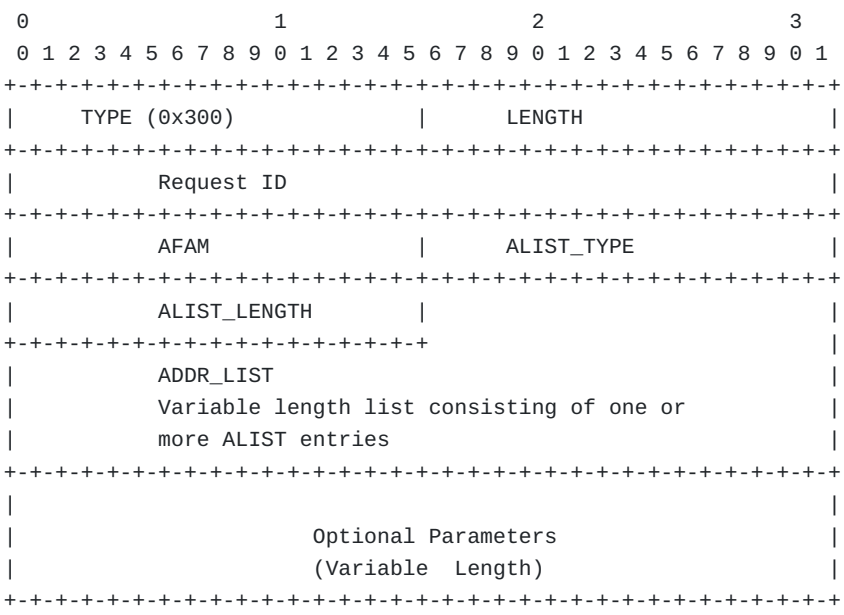
A TSR receiving a TDP_PIE_REQUEST_BIND must respond with a TDP_PIE_BIND or with a TDP_PIE_NOTIFICATION. A TSR that issues a TDP_PIE_BIND in response to a TDP_PIE_REQUEST_BIND places the Request ID from TDP_PIE_REQUEST_BIND in the Request ID field in the TDP_PIE_BIND that it issues.

When a TSR receiving a TDP_PIE_REQUEST_BIND is unable to satisfy it because of resource limitations it issues a TDP_PIE_NOTIFICATION for RESOURCE_LIMIT containing the Request ID from the TDP_PIE_REQUEST_BIND.

A TSR that issues TDP_PIE_NOTIFICATION with RESOURCE_LIMIT set must send a subsequent TDP_PIE_NOTIFICATION, containing the status notification RESOURCES, to the peer to whom it previously sent that TDP_PIE_NOTIFICATION when it has resources available to satisfy further TDP_PIE_BIND_REQUESTs from that peer.

If a TDP_PIE_NOTIFICATION is received containing RESOURCE_LIMIT the TSR may not issue further TDP_PIE_REQUEST_BINDs until it receives a TDP_PIE_NOTIFICATION with the Optional parameter RESOURCES.

This PIE has the following format:



TYPE:

Type field as described above. Set to 0x300 for TDP_PIE_REQUEST_BIND.

LENGTH:

Length in octets of the value field of this PIE. LENGTH is set to the length of the whole PIE in octets minus four.

Request ID:

This four octet unsigned integer contains a locally significant non zero value that a TSR uses to identify TDP_PIE_BINDs or TDP_PIE_NOTIFICATIONs that are generated in response to this request.

AFAM:

This 16 bit integer contains a value from ADDRESS FAMILY NUMBERS in Assigned Numbers [Reynolds] that encodes the address family that the network layer address in the tag bindings in the BINDING_LIST is from. This version of TDP supports IPv4 and IPv6.

ALIST_TYPE:

This 16 bit integer contains a value from the table below that encodes the format of the ALIST entries in the ADDR_LIST field. Currently there are 2 values defined by this specification.

ALIST_TYPE	ALIST entry format
0	Null list
1	Hop Count followed by variable length NLRI

The format for these entries is defined below.

ALIST_LENGTH:

Two octet unsigned integer that encodes the length in octets of the ADDR_LIST field.

ADDR_LIST:

A variable length list consisting of one or more entries of type ALIST_TYPE.

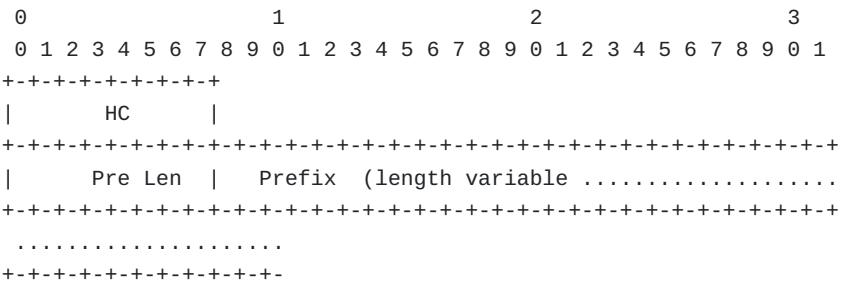
Optional Parameters:

This variable length field contains zero or more optional PIEs supplied in TLV structures.

4.7.1. ALIST formats

ALIST_TYPE = 0 indicates a null list ie there are no ALIST entries. A TDP receiving a TDP_PIE_REQUEST_BIND with ALIST_TYPE set to 0 interprets this as an implicit request for all the bindings that it currently has.

For ALIST_TYPE = 1 ALIST entries have the following form:



HC:

Hop count.

Pre Len:

This one octet unsigned integer contains the length in bits of the address prefix that follows.

Prefix:

A variable length field containing an address prefix whose length, in bits, was specified in the previous (Pre Len) field. A Prefix is padded with sufficient trailing zero bits to cause the end of the field to fall on an octet boundary.

4.7.2. Errors

Errors are reported using TDP_PIE_NOTIFICATION. If the TSR is unable to provide a TDP_PIE_BIND in response to a request the TSR indicates this by supplying the RESOURCE_LIMIT status notification as a parameter in the TDP_PIE_NOTIFICATION. The Request ID from the the TPD_PIE_REQUEST bind is supplied in the Value field of this status notification

STATUS NOTIFICATION	Type	Length	Value
RESOURCE_LIMIT	0x610	4	Request ID

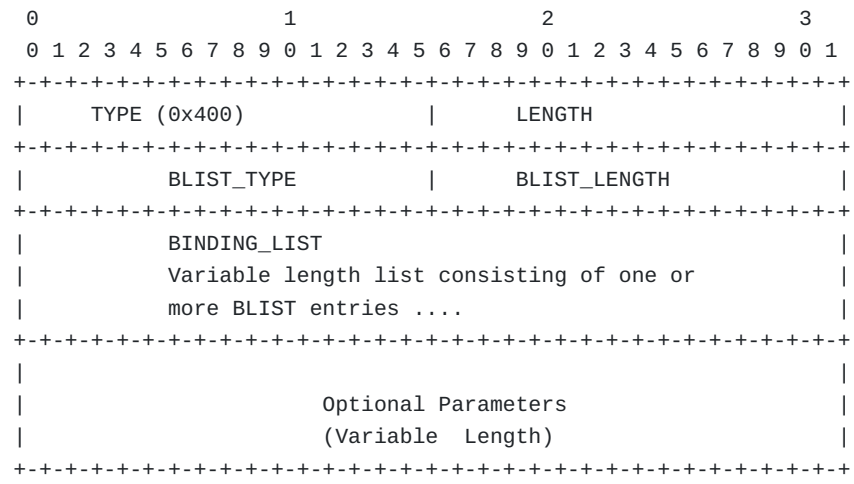
4.8. TDP_PIE_REMOVE_BIND

This PIE has two very different semantics which depend on whether the TSR issuing the TDP_PIE_REMOVE_BIND issued the TDP_PIE_BIND for the tag or whether it issued a TDP_PIE_REQUEST_BIND that resulted in it receiving a TDP_PIE_BIND for the tag. The second of these two cases is one of downstream allocation on demand. The rules for these two cases are:

If TDP_PIE_REMOVE_BIND is issued by the TSR that originally provided a binding containing the tag in question it is an absolute instruction to the TSR that receives it that it may not continue to use that tag to forward traffic to the TSR that issued it.

If TDP_PIE_REMOVE_BIND is issued by a TSR that received the tag as a consequence of an Upstream Request/downstream assignment sequence then it is an indication to the TSR that receives it that the TSR that requested the binding no longer needs it.

This PIE has the following format.



TYPE:

Type field as described above. Set to 0x400 for TDP_PIE_REMOVE_BIND.

LENGTH:

Length in octets of the value field of this PIE. LENGTH is set to the length of the whole PIE in octets minus four.

BLIST_TYPE

This 16 bit integer encodes the format of the BLIST entries in the BINDING_LIST field. Possible values are defined in Section 4.6. A TDP receiving this PIE with the BLIST_TYPE set to Null interprets it (based on the semantics) as either (a) an implicit instruction to remove all bindings belonging to the peer that issued the PIE, or (b) as an indication that all the bindings requested by the peer are no longer needed by the peer that issued the PIE.

BLIST_LENGTH:

This 16 bit unsigned integer encodes the length in octets of the BINDING_LIST.

BINDING_LIST:

Variable length field consisting of one or more BLIST entries of the type indicated by BLIST_TYPE. The format of these entries is defined in Section 4.6.

Optional Parameters:

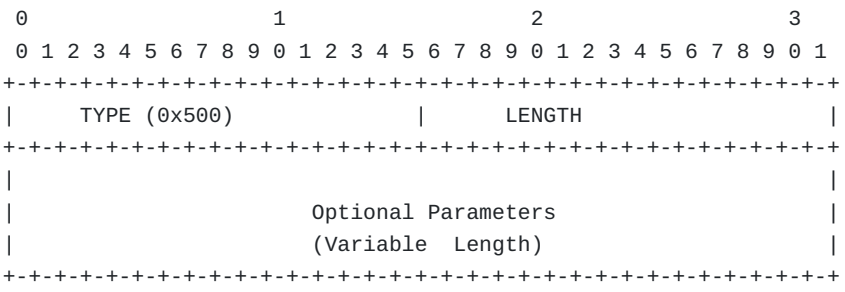
This variable length field contains zero or more optional PIEs supplied in TLV structures.

4.9. TDP_PIE_KEEP_ALIVE

The Hold Timer mechanism described earlier in Sections 3 and 4 is reset every time a TDP_PDU is received. TDP_PIE_KEEP_ALIVE is provided to allow reset of the Hold Timer in circumstances where a TDP has no other information to communicate to its peer.

A TDP must arrange that its peer sees a TDP_PDU from it at least every HOLD_TIME period. That PDU may be any other from the protocol or, in circumstances where there is no need to send one of them, it must be TDP_PIE_KEEP_ALIVE.

This PIE has the following format



TYPE:

Type field as described above. Set to 0x500 for TDP_PIE_KEEP_ALIVE.

LENGTH:

Length in octets of the value field of this PIE. LENGTH is set to the length of the whole PIE in octets minus four. LENGTH must be at least 2 for TDP_PIE_KEEP_ALIVE.

Optional Parameters:

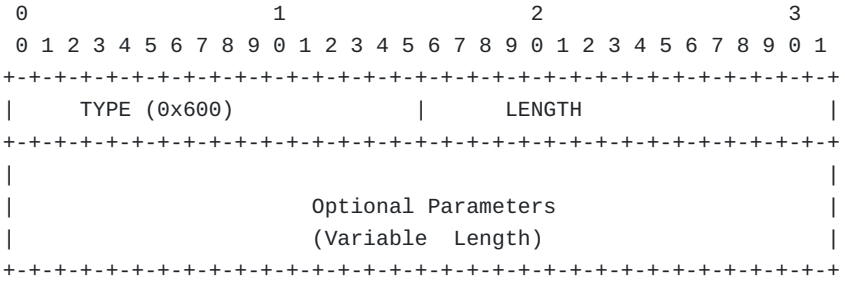
This variable length field contains zero or more optional PIEs supplied in TLV structures.

4.10. TDP_PIE_NOTIFICATION

TDP_PIE_NOTIFICATION is issued by TDP to inform its peer of a significant event. 'Significant events' include errors and changes in TSR capabilities or operational state.

All notification information is encoded as TLVs in the optional parameters field.

This PIE has the following format



TYPE:

Type field as described above. Set to 0x600 for TDP_PIE_NOTIFICATION

LENGTH:

Length in octets of the value field of this PIE. LENGTH is set to the length of the whole PIE in octets minus four.

Optional Parameters:

This variable length field contains zero or more optional parameters supplied in TLV structures. Optional parameters are defined for ERROR, STATUS and operational change notification.

The optional parameter types and their uses are:

ERROR:

There is only one error defined at this time, RETURNED_PDU. A TSR uses this Error to return a PDU to the TSR that issued it.

ERROR	Type	Length	Value
RETURNED_PDU	0x601	Var	Peer's PDU

As much as possible of the complete PDU, including the header, that is to be returned is inserted into the value field. The Length is set to the the number of octets of the PDU that is being returned that have been inserted into the Value field of this optional parameter. Implementations parsing this ERROR must be careful to recognize that the returned PDU may have been truncated.

STATUS:

The following STATUS notifications are defined:

STATUS NOTIFICATION	Type	Length	Value
RESOURCE_LIMIT	0x610	0	0
RESOURCES	0x611	0	0

The use of RESOURCE_LIMIT and RESOURCES are described in section 4.7.

OPERATIONAL:

The following notification of OPERATIONAL change is defined:

OPERATIONAL NOTIFICATION	Type	Length	Value
CLOSING	0x630	0	0

TDP may send a TDP_PIE_NOTIFICATION with CLOSING set in response to a protocol error or to administrative intervention.

A TDP receiving or issuing this notification transitions to the INITIALIZED state.

5. Intellectual Property Considerations

Cisco Systems may seek patent or other intellectual property protection for some or all of the technologies disclosed in this document. If any standards arising from this document are or become protected by one or more patents assigned to Cisco Systems, Cisco intends to disclose those patents and license them on reasonable and non-discriminatory terms.

6. Acknowledgments

We acknowledge with thanks the efforts of those people in cisco who reviewed earlier drafts of this document.

7. References

[Deering] Deering, S. et al "An Architecture for Wide Area Multicast Routing", Pro Sigcomm 94 in Computer Communications Review Vol 24 No 4.

[Rekhter] Rekhter, Y. et al "draft-rfced-tag-switching-overview-00.txt".

[Reynolds] J Reynolds, J Postel. "Assigned numbers" RFC 1700, October 1994

8. Author Information

Paul Doolan
cisco Systems, Inc.
250 Apollo Drive.
Chelmsford, MA 01824
Phone: (508) 244-8917
email: pdoolan@cisco.com

Bruce Davie
cisco Systems, Inc.
250 Apollo Dr.
Chelmsford, MA 01824
email: bsd@cisco.com

Dave Katz
cisco Systems, Inc.
170 Tasman Dr.
San Jose, CA 95134
email: dkatz@cisco.com

Yakov Rekhter
cisco Systems, Inc.
170 Tasman Dr.
San Jose, CA 95134
email: yakov@cisco.com

Eric Rosen
cisco Systems, Inc.
250 Apollo Dr.
Chelmsford, MA 01824
email: erosen@cisco.com