Network Working Group
Internet-Draft
Intended status: Informational
Expires: May 21, 2008

D. Jen
M. Meisel
D. Massey
L. Wang
B. Zhang
L. Zhang

November 18, 2007

APT: A Practical Transit Mapping Service
draft-jen-apt-01.txt

Status of this Memo

Copyright Notice

Abstract

The size of the global routing table is a rapidly growing problem. Several solutions have been proposed. These solutions commonly divide the Internet into two address spaces, one for determining the delivery location, and one to use during transit. Packets destined for delivery addresses are tunneled through the default-free zone

(DFZ), which uses only transit addresses. For this process to work, there must be a mapping service that can supply an appropriate destination transit address for any given delivery address. We present a design for such a mapping service. We adhere to a "do no harm" design philosophy: maintain all desirable features of the current architecture without negatively affecting its security or reliability. Our design aims to minimize delay and prevent loss in packet encapsulation, minimize the number of modifications to existing hardware, minimize the number of new devices, and keep the level of control traffic manageable.

Table of Contents

1. Requirements Notation

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

2. Problem Statement

The unexpected, explosive growth of the Internet is causing a greater and greater strain on its infrastructure. This problem has been well-documented in [RAWS][AddrAlloc]. Several solutions have been proposed to address this problem [CRIO][EFIT][EFIT-ID][LISP][SixOne] the majority of which involve separating the Internet into two parts, one for determining the delivery location, and one to use during transit. Routers in transit space would only need to know how to route to transit prefixes, which are stable and conducive to topological aggregation. When a packet is sent from source delivery address A to destination delivery address B, A's provider-edge router (the ingress tunnel router, or "ITR", as defined in [LISP]) encapsulates the packet and sends it through transit space to B's provider-edge router (the egress tunnel router, or "ETR"). B's ETR decapsulates the packet and forwards it to the appropriate recipient, B.

When encapsulating a packet, A's ITR must somehow determine B's ETR's transit address and include it in the outer header. In general, any ITR must be able to map any given delivery address to a corresponding ETR transit address for proper tunneling through transit space. This illustrates the need for a mapping service that can provide this address. The design details of this mapping service will play a large part in determining the effectiveness of any proposed implementation of a delivery/transit address space separation. The mapping service also presents a new opportunity to enhance the services currently offered by the Internet, which is further reason to carefully consider how this service should be implemented. Should mapping information be distributed via a push or a pull model? What additional information, if any, should be distributed along with the mapping information? Can we satisfy the mapping requirement without impacting packet delivery quality?

Our answers to these questions are rooted in a "do no harm" design philosophy: improve routing scalability without sacrificing any desirable features in the current architecture or negatively affecting its security and reliability. To this end, we present APT, a practical transit mapping service designed with the following goals in mind.

- Minimize delay and prevent loss in packet encapsulation.

- Minimize the number of devices that need to be modified to support APT.

- Minimize the number of devices that will require additional resources or complexity.

- Keep the design modular so that the method used to propagate mapping information is independent from the method used to retrieve mapping information for tunneling.

3. Terminology

Transit Network (TN) - An AS whose business is to provide packet transport services for its customers. Transit networks provide packet forwarding services for delivery networks (see definition below). As a rule of thumb, if the AS appears in the middle of any ASPATH in a BGP route today, it is considered a transit network.

Delivery Network (DN) - A network that is a source or destination of IP packets, but forwards packets between neither TNs nor other delivery networks.

Transit Space - The IP address space used by transit networks. We will also use the term "transit space" to refer to the topological area of the Internet where transit addresses are routable.

Delivery Space - The set of all IP address spaces used by delivery networks. We will also use the term "delivery space" to refer to the topological area of the Internet outside of transit space -- that is, where only delivery addresses are routable.

Transit Address (Taddr) - A Taddr is an address in transit space.

Delivery Address (Daddr) - A Daddr is an address in delivery space.

Default Mapper - A new device required by APT. Each transit network MUST have at least one default mapper. A default mapper maintains a complete mapping table. In other words, given any Daddr, default mappers can return a corresponding Taddr. To support the growing trend towards multihoming, the mappings stored in default mappers will map a Daddr prefix to a non-empty SET of destination Taddrs, all of which are expected to have a direct connection to the DN.

Tunnel Router (TR) - All edge routers in a TN will become TRs. Like ITRs and ETRs in LISP [LISP], TRs provide the encapsulation and

decapsulation services required for tunneling packets through transit space. A TR has both ITR and ETR functionality, meaning that any TR can perform both encapsulation and decapsulation of packets. To properly encapsulate any given packet, TRs can query the default mappers for mapping information. TRs also cache commonly used MapRecs locally. Note that TR cache entries are NOT identical to the mappings stored at default mappers (see the definitions of "MapSet" and "MapRec" below).

APT Node - A general term referring to any device type introduced by APT. This includes both default mappers and TRs.

MapSet - A MapSet contains a Daddr prefix and a non-empty SET of ETR Taddrs associated with the prefix. MapSets also include related information such as priority rankings for each of the ETRs in the set. Default mappers store MapSets.

MapRec - A MapRec contains a Daddr prefix and any SINGLE ETR Taddr associated with that prefix. Any MapRec is a subset of the complete MapSet for its Daddr prefix. TRs store MapRecs along with an associated TTL. A MapRec is removed from a TR's cache once its TTL expires.

4. APT Overview and Requirements

This section is a comprehensive overview of the devices and protocols introduced by APT. For explanations and justifications, see the corresponding referenced sections.

Default Mapper Requirements (see Section 5)

- Default mappers must have enough storage space to store the full, global mapping table and associated metadata.

- Every destination Taddr in a MapSet MUST have an associated time before retry (TBR, see Section 6.1).

- Default mappers MUST keep track of the Taddrs of the TRs they serve.

- Default mappers MUST examine the destination Taddr of incoming packets for addresses other than their own.

TR Requirements (see Section 5)

- TRs MUST keep a small cache to hold recently-used MapRecs and their TTLs.

- TRs MUST have a default route to their default mapper.

- TRs MUST be able to encapsulate and decapsulate IP-in-UDP packets with an APT header (see Section 11).

Failover for Multihomed DNs (see Section 6.1)

- When a Taddr prefix is withdrawn via BGP (see Section 6.1.1)

  * ITRs forward packets destined for unroutable Taddrs to their default mapper.

  * The default mapper forwards the packet to an alternate ETR if one is available.

  * The default mapper sends a Cache Add Message to the originating ITR.

- When a TR becomes unreachable (see Section 6.1.2)

  * Packets destined for the TR are intercepted by its default mapper.

  * The default mapper sets the TBR for the appropriate MapRec.

  * The default mapper forwards TR-addressed packets to an alternate ETR if one is available.

  * The default mapper sends an ETR Unreachable packet to the ITR's default mapper.

  * The default mapper broadcasts a Cache Drop Message to its TRs.

  * The ITR's default mapper sets the TBR for the appropriate MapRec.

  * The ITR's default mapper broadcasts a Cache Drop Message to its TRs.

- When a DN becomes unreachable from its TR (see Section 6.1.3)

  * The TR forwards packets destined for the DN to its default mapper, setting the APT packet type to ETR-to-DN link failure (see Section 11.1).

  * The default mapper sets the TBR for the appropriate MapRec.

   - The default mapper forwards the packet to an alternate ETR if one is available.

   - The default mapper sends a Delivery Network Unreachable packet to the ITR's default mapper.

   - The default mapper broadcasts a Cache Drop Message to its TRs.

   - The ITR's default mapper sets the TBR for the appropriate MapRec.

   - The ITR's default mapper broadcasts a Cache Drop Message to its TRs.

Mapping Dissemination

- Default mappers MUST sign updates with their TN's private key.

- Default mappers MUST verify the signature before processing or forwarding MapSet updates (see Section 8).

- Default mappers MUST NOT remove or alter the signature when forwarding the update.

- Default mappers MUST cryptographically sign control messages that may need to travel between ASes.

- Default mappers MUST speak DM-BGP and peer with other default mappers (see Section 7.1).

   - DM-BGP is a separate instance of standard BGP that runs on a different TCP port.

   - Only default mappers speak DM-BGP.

   - DM-BGP updates carry mapping updates in a new attribute type.

5. The Mapping Service

TRs serve as the gateway between delivery and transit space. When a TR receives a packet from a DN that needs to be routed through transit space, it maps the packet's destination Daddr to an appropriate destination Taddr (the mapping lookup details are presented below). The TR will then encapsulate the packet with a UDP header containing an APT header followed by the original layer-3 packet as the UDP payload (see Section 11). The packet can then be routed through transit space.

To minimize the latency introduced by encapsulation, APT seeks to store mapping information as close to the ITR as possible. However, the global mapping table is likely to grow very large over time. To avoid undue memory requirements for ITRs while still keeping mapping information within reach, we introduce the concept of default mappers.

A TR does not need to store the entire global mapping table. Instead, it queries a default mapper for mapping information and caches recently used MapRecs.

Default mappers are the only devices in the network that need to store the complete global mapping table. As we will see in the following example, TRs only make use of default mappers in the event of a cache miss. This means that, given sufficiently sized caches at the TRs, network latency will not heavily depend upon default mapper performance. Note that each TN need only have a single default mapper, but may choose to deploy more to avoid a single point of failure and to enhance overall performance. In the latter case, a TN MAY choose to use anycast to reach one of the default mappers or use multicast to reach all of them.

5.1. A Mapping Example

Below is a simple topology for demonstrative purposes. A and B are DNs, each addressable via a single Daddr prefix, TN1 and TN2 are TNs, ITR1, ETR1, and ETR2 are TRs, any node labeled "X" is a router, and M1 and M2 are default mappers. A portion of the mapping table for M1 is shown.

```
        ___                               ___
       / A \                             / B \_________
       \___/                             \___/        |   Delivery Space
- - - - -|- - - - - - - - - - - - - - - - -| - - - - -|- - - - - - - - -
      .--+---.                          .--+---.      |    Transit Space
   __-| ITR1 |-__                    __-| ETR1 |-__   |
  /   '------'  .`--.            .--'.  '------'  .`--+--.
 |   T   ____   | X |------------| X | T   ____   | ETR2 |
 |   N  | M1 |  '-;-'            '-:-' N  | M2 |  '-;----'
  \  1  '-/\-'   /                  \  2  '----'   /
   \_____/  \___/                    \____________/
 _______/    \___________________
|    DN    | TS Addr  | Priority |
|----------|----------|----------|
|   ...    |   ...    |    ...   |
|----------|----------|----------|
|    B     |   ETR1   |    10    |
|          |   ETR2   |    20    |
|----------|----------|----------|
|   ...    |   ...    |    ...   |
'--------------------------------'
```

Figure 1

In this section, we illustrate how TRs and default mappers interact within a TN to properly tunnel packets through transit space.

In Figure 1, a node in network A sends a packet to a Daddr in network B. When this packet arrives at ITR1, ITR1 looks up the destination Daddr in its MapRec cache. If a matching prefix is present in its cache, ITR1 simply encapsulates the packet with the corresponding destination Taddr and sends it across transit space. If a matching prefix is not present, ITR1 will send the packet through its default mapper, M1. It does this by encapsulating the packet with the (possibly anycast) address for its default mapper(s) as the destination Taddr.

This packet will arrive at M1, the only default mapper in TN1. When M1 receives the packet, it decapsulates the packet and examines the destination Daddr. Since default mappers store the full, global mapping table, a default mapper will always be able to encapsulate the packet with a valid destination Taddr. All packets encapsulated

by a default mapper MUST contain the default mapper's Taddr as the source address.

In addition to forwarding the packet to an appropriate TR (ETR1, in this case), M1 also treats the incoming packet as an implicit request from ITR1 for mapping information. M1 responds to ITR1 with a Cache Add Message (see Section 11.2) containing a MapRec that maps B to ETR1. This allows ITR1 to add this MapRec to its cache so that ITR1 can tunnel further packets destined for B directly to ETR1. The MapRec also has an associated time to live (TTL) that is set by M1. The TTL ensures that ITR1 will occasionally re-request this mapping information from M1. At this time, if the mapping information has changed in any way since ITR1's prior request, M1 can respond with an updated MapRec. Without this TTL, ITR1's cached information may become stale over time.

6. Multihoming Support

In the example above, the observant reader may have noted that B is multihomed. That is, B can be reached through both ETR1 and ETR2. Multihoming provides B with both enhanced reliability in case of a connectivity failure and the flexibility to distribute incoming traffic across different tunnel endpoints.

In accordance with our design goals, all of the logic for selecting a tunnel endpoint for a multihomed DN is contained within default mappers. Default mappers store full MapSets containing the addresses of all ETRs for a given Daddr prefix, while TRs only store a single MapRec per Daddr prefix. When a TR requests a MapRec for a multihomed DN, it is up to the default mapper to decide which one to return.

Many DNs will want to have some control over which tunnel endpoint is used for incoming traffic. Therefore, each MapRec in a MapSet has an associated priority value, which is made available to all default mappers throughout the transit space (see Section 7). The number is to be treated like a ranking -- an ETR with a lower priority value is more preferable.

At the same time, a sending TN may have its own preference regarding which of the ETRs to use for a given Daddr prefix. Default mappers can use a combination of locally configured routing policies and MapSet priority information to choose from the set of valid ETR addresses. Going back to Figure 1, assume that ITR1 does not have a MapRec for B in its cache. When A addresses a packet to B, ITR1 will send the packet to M1. If M1 has no preference between ETR1 and ETR2, it will examine the priority values in B's MapSet and select

ETR1, B's most preferred ETR. M1 forwards the packet to ETR1 and returns the corresponding MapRec to ITR1, which stores the MapRec in its cache.

In the case of a priority value tie, the default mapper can break the tie by picking the ETR to which it has the shortest path. If some ETRs are tied in terms of both lowest priority value and shortest path, the default mapper is free to break the tie arbitrarily. The address of the selected ETR will be used as the destination address when encapsulating the packet.

We envision that DNs will be able to manipulate their incoming traffic load by setting appropriate priority values in their MapSet. A DN who wants load balancing can assign the same priority value to all of his MapRecs. A DN who wants to have one TN as a primary provider and another only as a backup can simply assign a higher priority value to his ETR at his backup provider.

6.1. Using Alternate ETRs During Failures

When a network failure has rendered an ETR unable to perform its duties, an affected multihomed user will expect his traffic to be temporarily routed through an alternate ETR. There are three general types of failures that would require an ITR to use an alternate ETR: (1) an ITR may have discovered via BGP that it can no longer reach the Taddr prefix containing the address of the intended ETR, (2) the ETR itself may go down or lose connectivity, and (3) the link between a DN and its TR may be down, a new problem introduced by the tunneling architecture. This section will explain how each type of failure is handled, using Figure 1 as a reference. We assume that, at the time of failure, all TNs are using ETR1 to reach B.

To assist in handling these failures, default mappers store a time before retry (TBR) for each MapRec. Normally, the TBR for each MapRec is set to zero, indicating that it is usable. Any MapRec with a non-zero TBR value is considered invalid. We will refer to the action of setting a MapRec's TBR to a non-zero value as "invalidating a MapRec." MapRecs that map to unroutable destinations are also considered invalid. So long as a MapRec is invalid, default mappers will not use this entry as a destination address or include it in mapping responses. The role of the TBR in handling failures will become clear in the explanations below.

6.1.1. Handling Taddr Prefix Failures

For failures of type (1), ITR1 has no route to ETR1. Assume a host in network A attempts to send a packet to a host in network B. If ITR1 does not have a MapRec for B in its cache, it will forward the

packet to M1 (see Section 5.1). If ITR1 does have a MapRec for B in its cache, it will see that it has no route to ETR1, and forward the packet to its default mapper, M1. M1 will also see that it has no route to ETR1, and thus select the next-most-preferred ETR for B, ETR2. If it has a route to ETR2, it sends the packet with ETR2 as the destination Taddr and replies to ITR1 with the corresponding MapRec. M1 can assign a relatively short TTL to the MapRec in its response. Once this TTL expires, ITR1 will forward the next packet for B to the default mapper, which will respond with the most-preferred MapRec that is routable at that time. This allows ITRs to quickly revert to using ETR1 once it becomes reachable again.

6.1.2. Handling Single-ETR Failures

In the second case, the Taddr prefix containing ETR1 is still routable from ITR1, but ETR1 has failed or is otherwise unreachable. Since this failure is confined to TN2, all routers in TN2 should be able to detect that ETR1 is unreachable via TN2's IGP. In order to prepare for this situation, M2 announces a very high-cost link to all of the TRs it serves (in this case, ETR1 and ETR2) via IGP. When ETR1 fails, since the normal IGP path to ETR1 will no longer be valid, all packets addressed to ETR1 will be forwarded to M2 instead.

When M2 receives a data packet addressed to one of the TRs it serves (ETR1, in this case), it will assume the TR is unreachable, invalidate the corresponding MapRec, and broadcast a Cache Drop Message (see Section 11.3) to all of the TRs it serves. Using the default mapper address in the APT header (see Section 11), it will also reply to the sender's default mapper (in this case, M1) with an ETR Unreachable Message (see Section 11.4). M1 can then also invalidate the corresponding MapRec and broadcast a Cache Drop Message to its TRs.

In order to minimize packet losses, M2 should not simply drop data packets addressed to ETR1. Instead, M2 should attempt to reroute the packet to an alternate ETR, even if that ETR is in a different TN. It can do this by simply decapsulating the packet, looking up the MapSet for the Daddr prefix, and re-encapsulating the packet with a valid ETR as the destination Taddr according to the normal ETR-selection guidelines.

6.1.3. Handling TR-to-DN Link Failures

The final case involves a failure of the link connecting ETR1 to B. When ETR1 discovers it cannot reach B, it will send packets destined for B to its default mapper, M2, setting the APT message type to ETR-to-DN Link Failure (see Section 11.6) when encapsulating the packet. M2 will see that the packet's APT message type is ETR-to-DN Link

Failure, and handle this situation in the same way as situation 2 (see Section 6.1.2), except that the message it sends to M1 will be a DN Unreachable Message (see Section 11.5) instead of an ETR Unreachable Message.

DN Unreachable and ETR Unreachable Messages are handled the same way. However, we have kept them as separate notification types in order to allow for divergent behavior in the future.

7. Exchanging MapSets Between TNs

To avoid introducing latency or packet loss when encapsulating packets, the default mappers must have all MapSets available locally. In order for default mappers to store a full, global mapping table, there must be some way for them to receive MapSets from other TNs. However, only default mappers should receive MapSets. In this section, we propose a method for MapSet dissemination. The APT design in general does not depend on this particular method; it only requires that SOME method exists for secure, up-to-date, lightweight MapSet dissemination.

7.1. MapSet Dissemination via DM-BGP

MapSet dissemination can be accomplished using a separate BGP instance that is only run between default mappers. We refer to this new BGP instance as 'DM-BGP'. As a protocol, DM-BGP is identical to BGP, but it serves a different purpose. DM-BGP is used to disseminate MapSets, not as a reachability protocol. It is simply run on a different TCP port and is only used by default mappers so as not to affect the RIB-In of other nodes.

When a default mapper wishes to distribute his TN's mapping information to other default mappers, he sends out a DM-BGP update with the mapping information included as an optional, transitive BGP attribute with a new type. The NLRI included MUST be a prefix that uniquely identifies the source TN. When other default mappers receive DM-BGP updates, they store this information in their MapSet tables, replacing any existing MapSets. BGP policy knobs can still be tuned as desired by each TN. Upon receiving mapping updates, TNs can choose whether to forward the update to each of their peers, so long as their actions are in accordance with the BGP protocol.

A default mapper may receive the same mapping update more than once. This will occur when there is more than one DM-BGP path from the source default mapper's TN to the receiving default mapper's TN. Along with the mapping information, the new attribute should include a sequence number to allow receivers to detect duplicate mapping

updates. Default mappers MUST regularly announce MapSets to the rest of the network for all of the DNs to which their TN connects. As a precaution, however, these DM-BGP updates should be infrequent and rate-limited.

7.2. Regular MapSet Refresh

Regardless of the protocol used to disseminate MapSets, MapSets are not transient data. In order for default mappers to prevent their MapSet tables from strictly increasing in size without bound, they must be able to remove stale MapSets. For this reason, each MapSet entry MUST contain a time to live (TTL). A default mapper MAY remove a MapSet from its table at any time after this TTL has expired. In order to avoid premature removal from the global mapping table, default mappers MUST (1) regularly re-announce all MapSets for DNs they connect to and (2) set the TTL for each MapSet to no less than three times their refresh interval.

8. Security and Reliability

Using DM-BGP to distribute mapping announcements guarantees that they are only accepted from manually configured DM-BGP peers. This ensures that mapping updates are no less secure than routing updates are today. However, mapping updates have the potential to cause far more damage; with no security measures in place, a mapping update could direct ALL traffic for an entire Daddr prefix to an arbitrary Taddr. APT strives to prevent attacks and misconfigurations from having adverse effects outside of the TN in which they occur. Therefore, mapping updates will require some level of security.

8.1. Authenticating the Originator of Mapping Updates

Our first step towards authenticating mapping updates is to authenticate an update's originator. For this reason, each default mapper MUST cryptographically sign the mapping data in any update it originates. All default mappers within a single TN SHOULD use the same key pair, but default mappers in different TNs MUST use different key pairs. When a default mapper receives a mapping update, it MUST verify this signature before processing or forwarding the update. Default mappers MUST NOT remove or alter this signature when forwarding the update.

Clearly, this scheme can only work if there is a secure way to distribute all public keys to all default mappers. This should be a relatively straightforward problem to solve. We describe one simple, appropriate method for secure key distribution in a network of manually configured peers in a separate document (forthcoming).

8.2. Detecting MapSet Misconfigurations

Though the scheme outlined in Section 8.1 allows for secure authentication of the originator of a mapping update, it does not guarantee the correctness of the data. Since DM-BGP peerings are manually configured and therefore form a relatively closed network, misconfigurations are far more likely than attacks to be the cause of inaccurate mapping data.

The types of misconfigurations that could potentially be harmful are those that result in one TN accidentally interfering with the MapSet for a DN that it is not connected to. This can happen whenever a provider accidentally announces a MapSet for the wrong Daddr prefix. These types of accidental conflicts fall into three categories: (1) a TN announces a MapSet for the wrong Daddr prefix when that prefix already has a MapSet in the global mapping table, (2) a TN announces a MapSet for a Daddr prefix that subsumes a longer Daddr prefix that already has a MapSet, and (3) a TN announces a MapSet for a Daddr prefix that is a subset of a shorter Daddr prefix that already has a MapSet.

The first category of conflicts is the only one that we intend to actively prevent. Clearly, the DN that owns a particular Daddr prefix should be the ultimate authority for his mapping information. However, DNs do not announce their MapSet to the network directly, but rather through the TNs they connect to. In order to ensure a mapping update for a Daddr prefix is approved by its rightful owner, we must first include some sort of prefix owner identification in each MapSet. To this end, we introduce a DN key field into each mapping. This field SHOULD contain a cryptographically valid public key, but it is not currently used as such. When a default mapper receives a new MapSet that would replace an existing one, it only needs to ensure that the DN key has not changed. (This scheme is similar in spirit to the way that OpenSSH uses its 'known_hosts' file.) Note that DN keys are different from the keys used by default mappers to authenticate DM-BGP updates.

For the other two categories, it is less clear that such an announcement is the result of a misconfiguration. It is possible, for example, that the owner of a /16 Daddr prefix has resold some of the /24 prefixes it contains to other DNs. In such a case, only the administrators will know if the announcement is valid. It is for this reason that (in the spirit of PHAS [PHAS]) we do not attempt to prevent such changes, but only detect and notify interested parties. Since legitimate MapSet changes are infrequent, notifying interested parties of MapSet changes via e-mail is a perfectly viable option. These notifications could also prove useful in debugging the mapping service, or a particular TN's configuration.

8.3. APT Control Messages

APT never requires Cache Drop and Cache Add Messages to traverse AS boundaries. Any such message that does traverse an AS boundary must be an error or an attack. Therefore, TRs MUST ignore Cache Drop and Cache Add messages with a source Taddr outside of their TN. Since ISPs already generally drop packets from an external source when they contain a local source address, this simple policy should be sufficient to prevent TR cache poisoning, whether accidental or intentional.

Since any APT control message that may need to travel between ASes can also affect traffic flow, such control messages MUST be cryptographically signed. This currently includes ETR Unreachable Messages (see Section 11.4) and DN Unreachable Messages (see Section 11.5). Recall that the infrastructure required to generate and verify cryptographic signatures is already required for mapping update dissemination (see Section 8.1). When a default mapper receives such a control message, it MAY choose to verify this signature.

9. Scalability through Recursion

It is conceivable that the global mapping table could eventually grow large enough that it would no longer be possible to store it in a single default mapper. Theoretically, the global mapping table could grow to contain a separate MapSet for every Daddr prefix. In the case of IPv6 prefixes, the total number of MapSets would be on the order of 10^18, far more than we can expect to be able to store on a single device. If the global mapping table were to approach such gargantuan proportions, APT can simply be applied recursively.

In the recursive case, the terms "transit" and "delivery" are only meaningful relative to a particular depth of recursion, or number of times the packet has been encapsulated. We will refer to the non-recursive deployment of APT as the global level (G). What we have up until now referred to as delivery space and transit space are in fact G delivery space and G transit space.

At one level of recursion, G transit space is split into two address spaces: recursion depth 1 (R1) delivery space and R1 transit space. R1 delivery space is just another name for G transit space. Which name is used will depend on the context. R1 transit space can be further split into two R2 spaces, and so on. Using this terminology, all protocols and concepts in APT can be understood to apply generally at any level of recursion.

This figure shows the layout of a packet while being tunneled at an APT recursion depth of two.

```
 ________________________________________
|           R2 transit header            |
|----------------------------------------|
| R2 delivery a.k.a. R1 transit header   |
|----------------------------------------|
| R1 delivery a.k.a. G transit header    |
|----------------------------------------|
|           G delivery header            |
|----------------------------------------|
|                                        |
|               payload                  |
|                                        |
|________________________________________|
```

Figure 2

10. Mapping Announcements

Each mapping announcement has the following fields:

- Address Type - This field specifies the type of Daddrs used in the announcement. All Daddr prefixes in a single mapping announcement MUST be of the same address type. Currently, this is expected to be either IPv4 or IPv6, but other address types are also allowed.

- Total Length - This field specifies the total number of bytes used by all MapSets in the announcement. Each mapping announcement can contain MapSets for multiple prefixes, each with multiple MapRecs.

- Sequence Number - This field reflects the freshness of an update. Default mappers can avoid processing updates with old sequence numbers.

- Signature - The message should be cryptographically signed using the private key of the sending default mapper.

These fields are followed by one or more MapSets. Each MapSet in the announcement is described by the following fields:

- Daddr Prefix - This is the Daddr prefix for the MapSet.

- Time To Live (TTL) - This is the amount of time in hours that this MapSet should persist in default mappers before being considered obsolete and erased. This value MUST be set to at least three

times the sender's regular refresh interval. The TTL is specified in hours to prevent misconfigurations from causing excessive mapping updates.

- ETR Count - This is the total number of ETRs that the corresponding Daddr prefix maps to.

- Each ETR in a MapSet is described by the following fields:

  * Taddr - The address of this ETR.

  * Priority - Priorities are arbitrary integers that only have meaning in reference to each other. Taddrs with lower priority values are considered more preferable.

  * DN Public Key - This public key SHOULD uniquely identify the DN that owns this MapSet. It can be used to help identify configuration errors, and possibly for authoritative, cryptographic authentication of MapSet data in the future.

11. APT Header and Control Messages

Delivery space packets are encapsulated with a UDP header by an ITR. The UDP header should specify a well-known port reserved for APT, and the UDP payload MUST begin with an APT header. For regular data, a layer-3 header immediately follows the APT header. For other message types, we describe the fields that follow below.

11.1. APT Header Fields

The APT header contains the following fields:

- Version - The version of APT that should be used to interpret the header information.

- Tag - Extra field reserved for future use.

- Type - Determines the type of message being sent. Appropriate values are as follows:

  0: Regular Data

  1: Cache Add (Section 11.2)

  2: Cache Drop (Section 11.3)

3: ETR Unreachable (Section 11.4)

4: DN Unreachable (Section 11.5)

5: ETR-to-DN link failure (Section 11.6)

- Default Mapper Taddr - The address of the default mapper for the ITR that generated this header. This is the Taddr where any failure notifications from the destination TN will be sent. If this header was generated by a default mapper, this field SHOULD contain the same address as the source address in the encapsulating IP header.

11.2. Cache Add Messages

Cache Add Messages are only sent by default mappers to TRs within their own TNs, most notably in response to data packets. When a TR receives a Cache Add Message, it simply adds the enclosed MapRec to its cache, replacing any existing cache entry.

- Daddr Prefix - This is the Daddr prefix portion of the MapRec to be added to the receiving TR's cache.

- ETR Taddr - This is the Taddr portion of the MapRec to be added to the receiving TR's cache. It is the address of the ETR that can reach the Daddr prefix in the previous field.

- TTL - The TTL specifies the amount of time in seconds before the added cache entry should expire. Expired cache entries should be deleted from the TR's cache.

11.3. Cache Drop Messages

Cache Drop Messages are only sent by default mappers to TRs within their own TNs. When a TR receives a Cache Drop Message, it simply removes the cache entry corresponding to the enclosed Daddr prefix from its cache, if such an entry exists.

- Daddr Prefix - This is the Daddr prefix of the MapRec to be dropped.

11.4. ETR Unreachable Messages

ETR Unreachable Messages are sent by default mappers to other default mappers to notify them of failures.

- Transit Address - This is the Taddr of the ETR that cannot be reached.

- o Signature - The message should be cryptographically signed using the private key of the sending default mapper.

11.5. DN Unreachable Messages

DN Unreachable Messages are sent by default mappers to other default mappers to notify them of failures.

- o Daddr Prefix - This is the Daddr prefix of the DN that cannot be reached.

- o Signature - The message should be cryptographically signed using the private key of the sending default mapper.

11.6. The ETR-to-DN Link Failure Message Type

This message type is used by an ETR for two purposes: (1) to indicate to its default mapper that its direct link to the DN for the enclosed data packet is down, and (2) to preserve that data packet so that the ETR's default mapper might deliver it to the DN by way of a different ETR.

12. Incremental Deployment

Incremental deployment methods and incentives for APT will be discussed in a separate draft (forthcoming).

13. IANA Considerations

This memo includes no request to IANA.

14. Security Considerations

Security considerations for APT are discussed in Section 8.

15. References

15.1. Normative References

[RFC2119] Bradner, S., "Key words for use in RFCs to Indicate Requirement Levels", BCP 14, RFC 2119, March 1997.

15.2. Informative References

[AddrAlloc]
Meng, X., Xu, Z., Zhang, B., Huston, G., Lu, S., and L. Zhang, "IPv4 Address Allocation and BGP Routing Table Evolution", ACM SIGCOMM Computer Communication Review (CCR) special issue on Internet Vital Statistics, Volume 35, Issue 1, p71-80.

[CRIO] Zhang, X., Francis, P., Wang, J., and K. Yoshida, "Scaling IP Routing with the Core Router-Integrated Overlay", Proc. International Conference on Network Protocols , 11 2005.

[EFIT] Massey, D., Wang, L., Zhang, B., and L. Zhang, "A Scalable Routing System Design for Future Internet", SIGCOMM IPv6 Workshop , 8 2007.

[EFIT-ID] Massey, D., Wang, L., Zhang, B., and L. Zhang, "A Proposal for Scalable Internet Routing and Addressing", Internet Draft, http://www.ietf.org/internet-drafts/draft-wang-ietf-efit-00.txt, 2 2007.

[LISP] Farinacci, D., Fuller, V., Oran, D., and D. Meyer, "Locator/ID Separation Protocol (LISP)", Internet Draft, http://www.ietf.org/internet-drafts/draft-farinacci-lisp-05.txt, 2007.

[PHAS] Lad, M., Massey, D., Pei, D., Wu, Y., Zhang, B., and L. Zhang, "PHAS: A Prefix Hijack Alert System", USENIX Security .

[RAWS] Meyer, D., Zhang, L., and K. Fall, "Report from the IAB Workshop on Routing and Addressing", Internet Draft, http://www.ietf.org/internet-drafts/draft-iab-raws-report-02.txt, 2007.

[SixOne] Vogt, C., "Six/One: A Solution for Routing and Addressing in IPv6", Internet Draft, http://www.ietf.org/internet-drafts/draft-vogt-rrg-six-one-00.txt.

Appendix A. Open Issues

MapSets contain a priority field for each ETR, but this does not allow for uneven distribution of traffic across ETRs with the same priority, e.g. a 75/25 split. To provide a mechanism for DNs to request such traffic distributions, we should also include a weight field for each ETR.

If a TN sends out inaccurate mapping announcements, other TNs can identify and respond to the misbehaving source TN. However, there are no preventative security measures in place. Is detection and response enough of a security measure?

We are considering automating customer-DN-to-provider-TN mapping updates. Under our current design, whenever a DN needs to update its mapping information (it may add, subtract, or change providers, or change its priority values), the DN must contact its provider TNs offline and request that they announce the updated mapping information. It is then up to the provider TNs to update the mapping information. As we have seen with DNS updates, human involvement introduces the possibility of human error and delay. We hope to provide DNs with an automated way to manage their mapping information.

Is it too much to ask ISPs to change all of their PE routers into TRs? We suspect that TR implementation should involve only software changes. Existing router hardware can do everything required by a TR. Thus, we suspect the cost should be reasonable.

Authors' Addresses

Dan Jen

Email: jenster@cs.ucla.edu

Michael Meisel

Email: meisel@cs.ucla.edu

Dan Massey

Email: massey@cs.colostate.edu

Lan Wang

Email: lanwang@memphis.edu

Beichuan Zhang

Email: bzhang@cs.arizona.edu

Lixia Zhang

Email: lixia@cs.ucla.edu

Full Copyright Statement

Intellectual Property

Acknowledgment

Funding for the RFC Editor function is provided by the IETF Administrative Support Activity (IASA).