APT: A Practical Tunneling Architecture for Routing Scalability*

Dan Jen
University of California
Los Angeles, CA
jenster@cs.ucla.edu

Michael Meisel
University of California
Los Angeles, CA
meisel@cs.ucla.edu

Daniel Massey
Colorado State University
Fort Collins, CO
massey@cs.colostate.edu

Lan Wang
University of Memphis
Memphis, TN
lanwang@memphis.edu

Beichuan Zhang
University of Arizona
Tucson, AZ
bzhang@arizona.edu

Lixia Zhang
University of California
Los Angeles, CA
lixia@cs.ucla.edu

ABSTRACT

The routing table has seen a rapid increase in size and dynamics in recent years, mostly driven by the growth of edge networks. This growth reflects two major limitations in the current architecture: (a) the conflict between provider-based addressing and edge networks' need for multihoming, and (b) flat routing's inability to provide isolation from edge dynamics. To address these limitations, we propose A Practical Tunneling Architecture (APT), a new routing architecture that enables the Internet routing system to scale independently from edge growth. APT partitions the Internet address space in two, one for the transit core and one for edge networks, allowing edge addresses to be removed from the routing table in the transit core. In order to tunnel packets between edge networks, APT provides an efficient mapping service between edge addresses and the addresses of their transit-core attachment points. We conducted an extensive performance evaluation of APT using trace data collected from routers at two major service providers. Our results show that APT can tunnel packets through the transit core by imposing a minimal delay on no more than 0.8% of all packets at the cost of introducing only one or a few new or repurposed devices per AS.

1. INTRODUCTION

As reported [18] at a recent workshop organized by the Internet Architecture Board (IAB), the Internet routing system is facing serious scalability problems fueled by a rapid increase in edge-site multihoming and traffic engineering. When edge sites multihome, their prefixes must be announced separately into the global routing table, defeating provider-based address aggregation. Many multihomed sites also split (*i.e.*, de-aggregate) their prefixes to load-balance incoming traffic through different providers. These trends are causing super-linear growth of the global routing table [2, 11, 17] and increasingly frequent routing updates, many from a small number of highly unstable edge sites [15, 21].

The scalability problem reflects a fundamental limitation of the current Internet routing architecture: the use of a single, inter-domain routing space for both transit provider networks and edge sites. A natural solution is to separate these two fundamentally different types of networks into different routing spaces. As estimated in [16], removing edge-site prefixes from the inter-domain routing system could reduce the global routing table size and update frequency by about one order of magnitude.

In addition to improved scalability, this separation can provide other benefits. End hosts will not be able to directly target nodes within the routing infrastructure, enhancing its security. Edge sites will enjoy benefits such as better traffic engineering and the ability to change providers without renumbering.

The idea of separating end customer sites out of inter-domain routing first appeared in [4, 10] more than a decade ago. It was named "Map & Encap" after the proposed process for bridging the two routing spaces: the source maps the destination address to a provider that serves the destination site, encapsulates the packet, and tunnels it to that provider. This idea started to attract attention from vendors and operators after the recent IAB report and has been actively discussed at the IRTF Routing Research Group. However, the original proposal was only an outline. It did not solve a number of important issues such as how to distribute the mapping information, how to handle failures, how to ensure security, and how to incrementally deploy the system.

In this paper, we present *APT (A Practical Tunneling architecture)*, a design for a concrete realization of the Map & Encap scheme that addresses all of these issues. APT uses a hybrid push-pull model to distribute mapping information, a data-driven notification mecha-

*UCLA Computer Science Dept. Technical Report #080004

nism to handle physical failures between edge sites and their providers, and a light-weight public-key distribution mechanism for cryptographic protection of control messages. APT can be deployed with little to no new hardware, and incurs minimal delay on no more than 0.8% of all packets, according to our trace-driven evaluation.

Note that separating provider and edge networks only redefines the scope of inter-domain routing; it does not change any routing protocols. Therefore, other efforts of designing scalable routing protocols, *e.g.*, compact routing [14] and ROFL [3], are orthogonal and are not affected by the change in architecture.

The remainder of this papers is organized as follows. Section 2 explains the Map & Encap scheme and the challenges to realizing it. Section 3 gives a high-level overview of our design and design principles. We describe the APT design in detail in Section 4 and present our evaluation results in Section 5. Section 6 outlines an incremental deployment plan. We discuss scalability, policy, and other issues in Section **??**. Finally, we present related work in Section 8 and conclude our paper in Section 9.

2. MAP & ENCAP OVERVIEW

Since APT is a realization of the Map & Encap scheme, we begin with an explanation of how Map & Encap works.

There are two types of networks in the Internet: *transit networks* whose business is to provide packet transport services for other networks, and *edge networks* that only function as originators or sinks of IP packets. As a rule of thumb, if the network's AS number appears in the middle of any AS path in a BGP route today, it is considered a transit network, otherwise it is considered an edge network. Usually ISPs are transit networks and end-user sites are edge networks. The IP addresses used by transit networks are called *transit addresses* and the IP addresses used by edge networks are called *edge addresses*. The corresponding IP prefixes are called *transit prefixes* and *edge prefixes*.

Map & Encap does not change any routing protocols. It changes the *scope* of routing by not announcing edge prefixes into the global routing system. In other words, the inter-domain routing protocol for transit networks maintains reachability information only to transit prefixes, resulting in smaller routing tables and fewer routing updates. To deliver packets from one edge site to another, border routers between the edges and the core need to tunnel the packets across the transit core, as illustrated in Figure 1. When a host in *Site1* sends a packet to a host in *Site2*, the packet first reaches *Site1*'s provider, *ISP1*. However, routers in *ISP1* cannot forward the packet directly to *Site2* since their routing tables do not have entries for any edge prefixes. Instead,

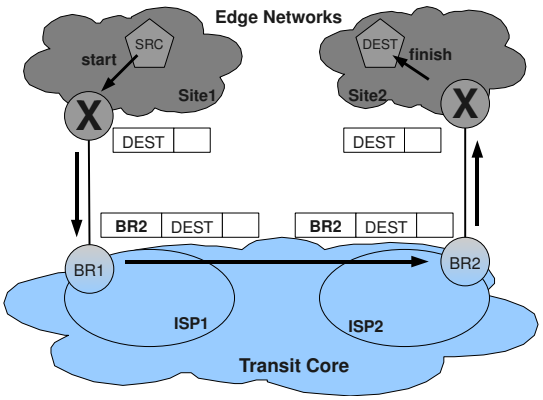

Figure 1: Separating Transit and Edge Networks

ISP1's border router, *BR1*, maps the destination address to *BR2*, a border router in *ISP2* that can reach *Site2*. Then the packet is encapsulated by *BR1*, tunneled through the transit core, decapsulated by *BR2* and delivered to *Site2*.

We call a border router that performs encapsulation when tunneling packets an *Ingress Tunnel Router (ITR)*, and one that performs decapsulation an *Egress Tunnel Router (ETR)*. A border router connecting a transit network to an edge network usually serves as both ITR and ETR, and can be referred to as a *Tunnel Router (TR)* in general. Internal ISP routers or routers connecting two ISPs do not need to understand the tunneling mechanism; they function the same way as they do today, only with a smaller routing table.

2.1 Challenges to Realization

There are a number of significant challenges that we must face when designing a practical realization the Map & Encap scheme. These challenges define a number of tradeoffs that must be kept in careful balance when developing a concrete design.

TR Placement.

In order to ensure all traffic is properly tunneled, a TR must be on the path between an edge network and its provider. Thus, we should pick the router at one end of the link connecting an edge network to its provider in the transit core. But which of these two routers should become a TR? From a technical standpoint, a provider-side router will generally serve many edge-side routers. As a result, there are fewer provider-side routers, but each one handles a greater quantity of traffic. From an economic standpoint, someone has to pay for the new infrastructure, but edge networks and transit networks have different incentives to do so.

Making Mapping Information Available at TRs.

Mapping information describes a relationship between a transit network and an edge network, which is not necessarily known by other parties on the Internet. To avoid a reduction in Internet service quality, it is important to minimize potential data loss and delay introduced by the extra step of retrieving this mapping information. Ideally speaking, if all mapping information were to be pushed to all ITRs, delay and loss would be minimal. However, the mapping table size would start with approximately the size of the current default-free zone (DFZ) routing table, and potentially grow quickly by one or two orders of magnitude. On the other hand, not equipping ITRs with the full mapping table would require pulling mapping information from a remote location. This implies a lookup delay, during which packets will incur additional latency and/or loss.

Scalability.

Since the main goal of Map & Encap is to solve the routing scalability problem, any realization of the Map & Encap scheme must itself be scalable. Due to the high cost of deployment, any change to the Internet architecture must be designed not to merely postpone the problem, but to counteract it as best we can.

Maintaining Reliability.

Today, the Internet often relies on the inter-domain routing protocol to discover failures in connectivity to edge networks. Once edge networks are removed from the transit core's routing tables, this method of discovering edge network failures will no longer be possible. Thus, a Map & Encap scheme must provide a new way to discover these failures if we intend to maintain the reliability of the current network.

Security.

Mapping can provide new opportunities to improve network security, but can also provide new opportunities for attackers to hijack or redirect traffic. A good design should exploit the former, and provide lightweight methods to prevent the latter.

Incremental Deployment.

On the Internet, one simply cannot set a flag day when all sites will switch to a new design, no matter how great an advantage the design offers. As a result, any design must explicitly assume incremental deployment. We must offer backwards compatibility for sites that are slow to adopt APT and also offer incentives for sites to adopt it.

3. APT OVERVIEW

We intend for APT to be a practical, deployable design for the real-world Internet. To ensure that our design meets this goal, we adhere to the following design principles.

- *Do no harm* to Internet services or service quality. Improve scalability while causing as little disruption as possible to current Internet services.
- *Align cost with benefit* by ensuring that no one is paying so that someone else can profit. We must acknowledge that the Internet infrastructure is owned by a number of independent entities that operate on a for-profit model.
- *Allow flexibility* for operators to make tradeoffs between performance and resources. We must acknowledge that the different administrative domains that make up the Internet will want to make such tradeoffs in different ways, and will only deploy a new system if it is flexible enough to allow this.

3.1 How APT Works

APT places TRs at the provider-side of the link between edge networks and their providers (see Figure 2). There are two main reasons for this, derived from our design principles. First, since Map & Encap is intended to solve the routing scalability problem and release the pressure on ISP routers, it is only natural that ISPs should pay the cost. This is one way in which APT aligns cost with benefit. Second, a tunnel has two ends, the ITR and the ETR. A solution should allow, but not require, both ends to be placed in the same administrative domain, such as within the network of a single ISP. This allows unilateral deployment of APT by a single ISP. Had we chosen to place TRs at the customer-side, no single edge network would be able to benefit from unilateral deployment.

To distribute mapping information, APT uses a hybrid push-pull model. All mapping information is pushed to all transit networks. However, within each transit network, only a small number of new devices called *default mappers (DMs)* store the full mapping table. ITRs store only a small cache of recently used mappings. When an ITR receives a data packet, it looks for an appropriate mapping in its cache. If such a mapping is present, it can encapsulate the packet and forward it directly to an appropriate ETR. Otherwise, it forwards the packet to a DM. The DM treats the packet as an implicit request for mapping information. In response, it sends an appropriate mapping to the requesting ITR, which stores the mapping in its cache. Meanwhile, the DM encapsulates and forwards the packet on behalf of the ITR. This process is illustrated in Figure 2.

Default mappers and tunnel routers have very different functionality. DMs are designed to manage the large mapping table, but only need to forward a relatively small amount of data traffic. TRs have small routing tables, but need to forward very large volumes

of traffic. This distinction will become even more prominent in the future as the Internet grows larger to include more edge networks and the traffic volume continues to increase. Since DMs and TRs are implemented in separate devices, both their hardware and software can be engineered for their specific purposes and both can scale appropriately for their specific tasks.

The association between an edge and a transit network may change due to either provider changes or border link failures. Provider changes occur when an edge network switches providers – an event that occurs in human time scale, likely measured in weeks or months. Physical failures of the links between transit and edge networks, however, can occur more frequently. In APT, only infrequent provider changes will trigger updates to the mapping table and be propagated to all transit networks. APT does not update the mapping table due to physical failures. Rather, APT takes a data-driven approach to edge-network unreachability notification. APT only informs those senders that are attempting to communicate with an unreachable edge network of the failure. This greatly reduces the scale of the physical failure's impacts.

By not storing the entire mapping table at every ITR, APT requires drastically less storage than a pure push model. By using data-driven local queries, APT mitigates the delay and prevents the loss associated with a pure pull model. By propagating the mapping table to all transit networks, APT allows individual networks the flexibility to manage their own mapping systems. A transit network can install more DMs to increase robustness and decrease latency, or fewer DMs to decrease the cost of deployment. By using data-driven failure notifications, APT notifies senders of edge-network unreachability while still eliminating the traffic caused by current edge-network routing updates. All of these design decisions honor our principles of doing no harm, aligning cost with benefit, and allowing for flexibility.

4. APT IN DETAIL

4.1 Default Mappers

In APT, a default mapper, or DM, performs the following functions.

- Maintaining the full mapping table. More specifically, it authenticates new mapping entries before accepting them, and removes entries that have exceeded their Lifetime value (see Section 4.5).

- Propagating mapping information to other DMs in neighboring ASes. DMs in different networks peer to form a DM mesh, via which mapping information is propagated throughout the entire transit core.

- Providing local ITRs with mapping information as needed. DMs provide a central management point for local traffic engineering policies. When an ITR requests mapping information, a DM can direct traffic by deciding which ETR address to provide in response.

- Forwarding packets in the event of an ITR cache miss.

- Handling transient failures without updating the mapping table. Only long-term changes such as provider changes will be reflected in the mapping table.

Although APT can work with just one DM in each transit AS, an AS may install multiple DMs for high robustness and load balancing, with each DM maintaining the full mapping table. To efficiently manage and communicate with multiple DMs, an AS configures an internal multicast group, DM_{all}, and an internal anycast group, DM_{any}. Packets sent to DM_{all} will be forwarded to all of the DMs in the same AS, and any router in the AS can reach the nearest DM by sending packets to DM_{any}. Thus, adding or removing DMs is transparent to other routers in the same AS.

Note that DM_{any} (DM_{all}) is an anycast (multicast) group local to a single AS. To prevent potential abuse, DM_{any} and DM_{all} are configured for internal use only. Any packet coming from outside of the AS destined to DM_{any} or DM_{all} will be dropped at the AS border. In the case that anycast is useful for external communication, a separate address, DM_{any_ext} is set up for external use. There is no multicast group for external use. If some external information needs to reach all DMs in an AS, it is always sent to one specific DM or to DM_{any_ext} for authentication and verification before being sent to DM_{all}.

4.2 Mapping Information

The mapping information in APT associates each edge prefix with one or more transit addresses, each belonging to an ETR in an ISP that serves the particular edge network. The ETR must have a direct connection to the edge network owning the prefix. For example, if a university owns the address prefix a.b/16 and has two Internet service providers ISP1 and ISP2, then a.b/16 will be mapped to the ETRs in ISP1 and ISP2 that directly connect to the university network.

To support traffic engineering, APT associates two values with each ETR address: a priority and a weight. When an ITR looks up the mapping information for an edge prefix, the ETR with the highest priority is picked. When multiple addresses have the same priority, they will be used in proportion to their weight. If an edge network wants to have one provider as a primary entry point for its incoming traffic and another as a backup, it

can simply assign a lower priority to the address(es) of the ETR(s) at its backup provider. If the network wants to load balance its incoming traffic between multiple providers, it can assign the same priority to multiple ETRs and use appropriate weights to split the traffic.

Mapping information for an edge prefix is generated in the following way. First, the edge network owning the prefix sends priorities and weights to each of its providers. Next, a default mapper in each provider announces a *MapSet* containing the edge prefix, its own ETR addresses for that prefix, and the edge network's priorities and weights.

Formally speaking, for an edge prefix p and its provider network N, MapSet(p, N) = $\{(d, w) \mid d$ is an ETR address in N **and** d is directly connected to p, and w is the priority and weight information for d $\}$. Note that one edge prefix may be mapped to multiple ETRs in the same provider network. If p is multihomed to m providers $N_1, N_2, ..., N_m$, MapSet(p) = $\bigcup_{i=1}^{m}$MapSet(p, N_i). To distinguish MapSet(p, N) from MapSet(p), we call the former a *Provider-Specific MapSet* and the latter a *Complete MapSet*, or simply a *MapSet*. Furthermore, we use the term *MapRec* to refer to the mapping from an edge prefix to any *single* ETR address.

4.3 Data Forwarding

Recall that an edge prefix's MapSet can contain many ETR addresses. When tunneling a packet to such a prefix, one of these ETR addresses must be selected as the tunnel egress. In order to keep TRs as simple as possible, we place all ETR selection logic in default mappers, including enforcement of the MapSet's priorities and weights. This allows ITRs to avoid any decision-making when forwarding high volumes of data and allows centralization of policy decisions.

To enable this, APT ITR caches contain only MapRecs. MapRecs contain mappings from an edge prefix to a *single* ETR address. When an ITR receives a packet from an edge network, it first tries to find a MapRec matching the destination address in its cache[1]. If the lookup is successful, the packet is tunneled from the ITR to the ETR address contained in the MapRec, just like in figure 1. When the ITR has a cache miss, it tunnels the packet to DM_{any}, the anycast address of the local DMs.

ITRs also maintain a *cache idle timer(CIT)* for each MapRec in their cache. The CIT for a MapRec is reset whenever the MapRec is accessed. Once a MapRec has been idle for an amount of time greater than the CIT value, the MapRec is flushed from the ITR's cache. The CIT is important for the performance of APT under edge-network reachability failures (see Section 4.4).

[1]In practice, the ITR would maintain a small BGP table and check this before the cache. This is done for backwards compatibility. See Section 6

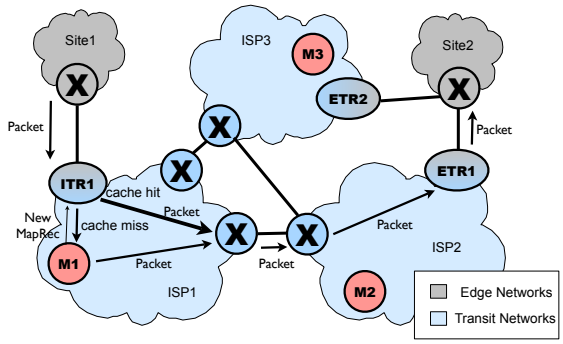

Figure 2: Example Topology for Data Forwarding

Upon receiving a tunneled packet from a local ITR, a DM first performs a longest-match lookup in its mapping table to find the MapSet for the destination address. It then selects one ETR address from the MapSet based on the priority, the weight value, and local policy. The DM then creates a MapRec and sends it to the ITR who sent the data packet. Other than the edge prefix and selected ETR address, the MapRec contains a CIT value assigned by the DM. Finally, the DM tunnels the packet to the selected ETR address, with the tunnel source address set to the original ITR.

Until the ITR receives the DM's response, it will continue to forward packets with the same destination prefix to the DM. The DM will continue to forward these packets, but will suppress duplicate control messages to the ITR using a *Deaf Timer* for the (ITR, edge prefix) pair. It will retransmit the MapRec only when the timer expires.

To illustrate the above process, Figure 2 shows a simple topology, where *Site1* and *Site2* are two edge networks, each owning edge prefix P_1 and P_2, respectively. *ISP1*, *ISP2* and *ISP3* are transit networks. A node in *Site1* sends a packet to a node in *Site2*. When this packet arrives at *ITR1*, it looks up the destination address d in its MapRec cache. There is no matching prefix, so *ITR1* sends the packet to a default mapper (*M1* in this case) by encapsulating the packet with DM_{any}(*ISP1*) as the destination address. When this packet arrives at *M1*, it decapsulates the packet and performs a longest-match lookup in its mapping table using the destination address d. Since d matches the prefix P_2, it will find the MapSet for P_2 containing *ETR1* and *ETR2*. *M1* selects *ETR1* based on the priority value, responds to *ITR1* with a MapRec that maps P_2 to *ETR1*, and then encapsulates the packet with *ETR1* as the destination address and sends it out.

4.4 Failure Detection and Recovery

In today's Internet, edge networks achieve higher re-

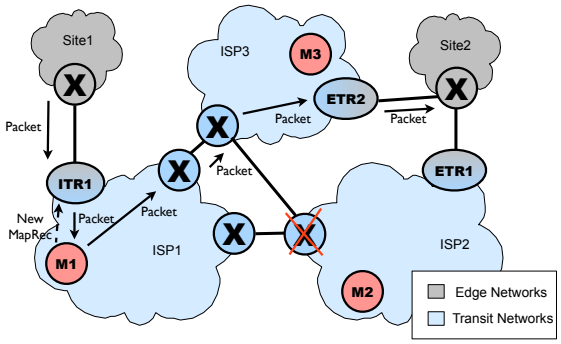

Figure 3: An example of a transit prefix failure.

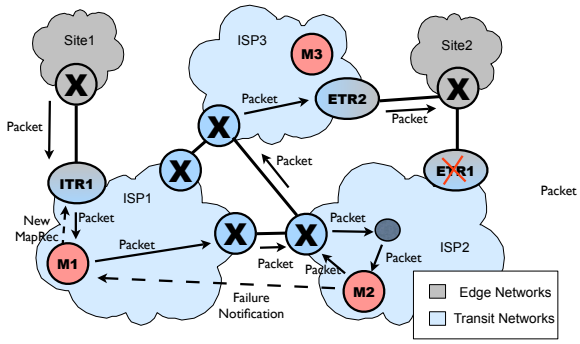

Figure 4: An example of a single ETR failure.

liability through multihoming. When connectivity to one provider fails, packets can be routed through other providers. Today, when such a connectivity failure occurs, this information is pushed into the global routing table via BGP. In APT, edge network connectivity is reflected in a mapping table that does not adjust to physical failures. Thus, an ITR may attempt to tunnel packets to an ETR that has failed or has lost connectivity to the edge network. APT must be able to detect such failures and route the affected traffic through an alternate ETR. Generally speaking, there are three types of failures that APT must handle:

1. The transit prefix that contains the ETR has become unreachable.
2. The ETR itself has become unreachable.
3. the ETR cannot deliver packets to the edge network. This can be due to a failure of the link to its neighboring device in the edge network, or a failure of the neighboring device itself.

4.4.1 Handling Transit Prefix Failures

An ITR will not necessarily be able to route traffic to all transit prefixes at all times. If an ITR attempts to tunnel a packet to an ETR in a transit prefix that it cannot currently reach, it treats this situation much like a cache miss and forwards the packet to a local default mapper. In Figure 3, *ITR1* has no route to *ETR1*, so it will 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 *Site2*, *ETR2*. It tunnels the packet to *ETR2* and replies to *ITR1* with the corresponding MapRec. *M1* can assign a relatively short CIT to the MapRec in its response. Once this CIT expires, *ITR1* will forward the next packet destined for *Site2* to a default mapper, which will respond with the most-preferred MapRec that is routable at that time. This allows *ITR1* to quickly revert to using *ETR1* once *ETR1* becomes reachable again.

4.4.2 Handling ETR Failures

When an ETR fails, packets heading to that ETR are redirected to a local DM in the ETR's transit network. This redirection is achieved through the intra-domain routing protocol (IGP); each DM in a transit network announces a high-cost link to all of the ETRs it serves. When an ETR fails, the normal IGP path to the ETR will no longer be valid, causing packets addressed to the ETR to be forwarded to a DM. The DM will attempt to find an alternate ETR for the destination prefix using its mapping table and tunnel the packet to that ETR.[2] The DM also sends an *ETR Unreachable Message* to the ITR's DM, informing the ITR's DM that the failed ETR is temporarily unusable. How the ETR's DM determines the ITR's DM address will be discussed in Section 4.5.2.

To avoid sending the address of an unreachable ETR to any subsequently requesting ITRs, default mappers also store a *Time Before Retry (TBR)* timer for each ETR address in a MapSet. Normally, the TBR timer for each ETR is set to zero, indicating that it is usable. When an ETR becomes unreachable due to a failure, its TBR timer is set to a non-zero value. The DM will not send this ETR address to any ITR until the TBR timer expires. We will refer to the action of setting a MapRec's TBR to a non-zero value as "invalidating a MapRec."

In Figure 4, traffic entering *ISP2* destined for *ETR1* should be directed to *M2*, the default mapper in *ISP2*,

[2]If the alternate ETR is in a different network, whether to forward packets in this situation is determined by the contractual agreement between the edge network and its providers.

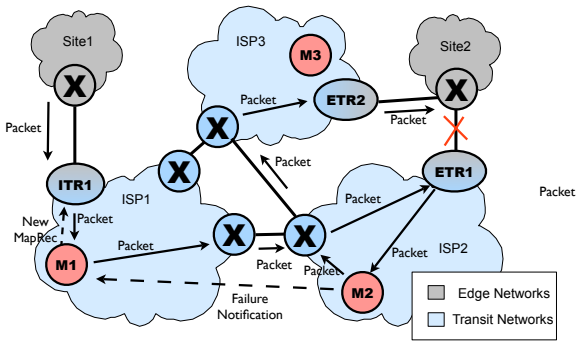

Figure 5: An example of a failure of the link connecting an ETR to its edge network.

according to *ISP2*'s IGP. When *M2* receives such a data packet, *M2* will tunnel the packet to *ETR2*, and notify *M1*, the default mapper in *ISP1*, of *ETR1*'s failure by sending an ETR Unreachable Message to $DM_{any\_ext}(Site1)$, the external anycast address for *ISP1*'s DMs (obtained via the Mapping Distribution Protocol, described in Section 4.5). *M1* can then send a new MapRec containing *ETR2* to *ITR1*. Similar to the previous case, the CIT for this MapRec will be relatively short.

4.4.3 Handling Edge Network Reachability Failures

The final case involves a failure of the link connecting an ETR to its neighbor in an edge network or the failure of the neighbor itself. This case is handled similarly to the previous case, except that the message sent to the ITR's default mapper will be of a different type, *Edge Network Unreachable*. In Figure 5, when *ETR1* discovers it cannot reach *Site2*, it will send packets destined for *Site2* to its DM, *M2*, setting the *Redirect Flag* when encapsulating the packet. The Redirect Flag signals to *M2* that the packet could not be delivered and should be re-routed. *M2* will redirect the packet to *ETR2* and then send an Edge Network Unreachable Message to *M1*.

4.5 Mapping Distribution Protocol

Making mapping information available to ITRs is one of the most important challenges in realizing a Map & Encap scheme. APT adopts a hybrid push-pull approach: it pushes the mapping information to DMs in all transit networks, but lets ITRs pull the mapping information from DMs.

4.5.1 DM Mesh

In APT, mapping information is distributed via a mesh of connections between DMs. These connections are configured manually based on contractual agreement, just as in BGP. Two neighboring APT ASes should establish at least one DM-DM connection between them. They can also choose to have multiple DM-DM connections for reliability. An AS can configure one or multiple DMs to connect to external DMs, but it is not required that all of its DMs have external connections. The DMs that have external connections will forward incoming mapping information to their local DM_{all} group, from which DMs without external connections will learn the mapping information.

Having the DM Mesh congruent to the AS topology facilitates incremental deployment and aligns maintenance and setup cost with benefit. Mapping information is just a small amount of additional data transmitted between two neighboring ASes that already have a contractual agreement for exchanging traffic. Since mapping exchange is bi-directional, it should benefit both parties equally. This means that both parties have incentives to maintain the connection well and fix any problems quickly.

4.5.2 The Dissemination Protocol

DMs exchange MDP messages using an OSPF-style flooding protocol, without the topology and path computation parts of OSPF. An MDP message has a header and a payload. Different payload types are supported. For mapping dissemination, the payload is provider-specific MapSets and the provider's DM_{any_ext} address. For security purposes, MDP is also used to propagate public keys and prefix lists for provider networks, which will be discussed in Section 4.6.

A DM originates MDP messages to push its own provider-specific MapSets to other provider networks. For instance, a customer network with prefix p is dual-homed through providers X and Y. Provider X's DM(s) would generate an MDP message containing $MapSet(p, X)$ and $DM_{any\_ext}(X)$ and send this message to its neighboring DMs. After this message propagates throughout the transit core, DMs in other networks will know the addresses of the ETRs in X's network via which prefix p can be reached. In case they need to send feedback information to X, they will use the address $DM_{any\_ext}(X)$ to reach X's DMs. Similarly, provider Y will announce $MapSet(p, Y)$ and its own $DM_{any\_ext}(Y)$. After receiving the provider-specific MapSets $MapSet(p, X)$ and $MapSet(p, Y)$, DMs combine them to get the complete MapSet for prefix p, including ETRs from both networks X and Y. Putting all MapSets together, a DM gets the complete mapping table to reach all edge prefixes.

The header of an MDP message contains control information necessary for efficient data dissemination. It includes (1) the AS number of the originator of the message, (2) a sequence number, and (3) a Lifetime. The

combination of the AS number and the sequence number uniquely identifies a message. It is used by a receiver to determine whether an incoming message is new. The Lifetime is used to make sure an outdated message will expire at certain time.

When a DM receives an MDP message from a neighboring DM, it will check whether this is a new message and make sure that the message has a Lifetime greater than one. Outdated, expired, or duplicate messages will be dropped. Accepted messages will be forwarded to all neighboring DMs except the one from which the message was received. Message transmission is acknowledged at every hop. The sending DM will retransmit the message if there is no acknowledgment from the receiving DM within certain time. The Lifetime is decremented as time goes by. Eventually, a MapSet will expire. It is the originating DM's responsibility to periodically re-generate its MDP messages to refresh other DMs. A DM can also explicitly withdraw its previous announcements by sending out a withdrawal message onto the DM mesh.

Since customer-provider relationships are usually stable for at least a month due to contractual obligations, the message Lifetime and the refresh frequency can be set to the scale of days or weeks, which means the volume of MDP traffic should be easily manageable. Other techniques in OSPF are also incorporated to help efficient dissemination. For instance, every time a DM reboots, it will synchronize its mapping table with its neighbor DMs to learn the most recent MapSets and sequence numbers.

4.6 Cryptographic Protection

While our design makes the global routing system more scalable and more flexible, we also need to make sure its security is not compromised. In answering this challenge, we intend to make APT as secure as the current Internet at least, making improving where practical.

APT adds new control messages that attackers could forge to manipulate packet forwarding. This constitutes a major security threat. For instance, a forged failover notification message could prevent ITRs from using certain ETRs, and a forged MapRec or MapSet could divert large quantities of traffic to arbitrary ETRs.

In APT, we add cryptographic protection to all control messages. We assume that every transit network has its own public-private key pair and signs all APT control messages that it generates. Receivers verify the signature before accepting a message. As in many other large scale systems, the main challenge in enabling cryptographic protection is how to distribute public keys in the first place. APT does not rely on a Public Key Infrastructure (PKI) for key distribution, since a PKI would require a significant amount of effort and coordination among all transit networks. The slow progress or lack of progress in deploying PKI-based solutions in the Internet (*e.g.*, DNSSEC and SBGP) suggests the need for an alternative that does not require a rigid delegation infrastructure.

4.6.1 Key Distribution

APT employs the DM Mesh to propagate every transit network's public key to all other networks in the transit core. To prevent attackers from forging someone else's public key, we require that every network have its neighbors verify and sign its key. For instance, if X has two neighbors, Y and Z, then X should have both neighbors verify X's public key and sign it. X will announce its key together with Y and Z's signatures through the DM Mesh. Similarly, X will also vouch for Y and Z's public keys.

Once every network announces its own key together with its neighbors' signatures, this information forms a web of trust, which a receiver can use to determine whether to trust a public key. For instance, assume X already trusts the public keys of networks Z and R. If X receives a message carrying W's public key and signatures from Z and R, then X can verify these signatures. If the two signatures indeed belong to Z and R, respectively, X will trust this message, record W's public key, and forward the message to its peers. Each network can configure its threshold for trusting a key, as long as this threshold is greater than one. Later, X can also use W's signature to verify other messages. If an attacker announces a false public key for W, he will not be able to forge the signatures of Z and R. In this case, X will discard the attacker's forged key.

Neighbor signatures are done when two neighbor ASes configure their DM connections. They verify the keys and signatures offline. Keys have a finite time-to-live after which they will expire. Keys can be replaced or revoked via a Rollover message or a Withdrawal message, respectively. These messages are signed by the old keys as well as the new keys if there are any. ASes should periodically rollover their keys, obtaining signatures from their neighbors for the new keys.

4.6.2 Attack Detection

Recall that APT adds cryptographic protection to all control messages. If private keys are compromised or networks misbehave, they can pose security threats that signatures cannot prevent. For instance, a misbehaving network, due to either operational errors or malicious acts, may inject mapping information for prefixes belonging to other networks, effectively hijacking other's traffic. This problem exists in the current Internet. In APT, we take advantage of the DM mesh and the flooding protocol to quickly detect such incidents, which is a significant improvement over the current Internet.

In APT, edge networks do not participate in the mapping dissemination process. However, they can still check the correctness of their mapping information by setting up an MDP monitoring session with their providers.[3] MDP ensures that a message will reach every provider network without changes. If there is an announcement of a false mapping for some edge prefix, the transit network(s) legitimately associated with that edge prefix will receive the message. Yet, since each provider only announces its own provider-specific MapSet, it cannot know whether another provider-specific MapSet for the same edge prefix is legitimate. A rogue network announcing a forged provider-specific MapSet for the same edge prefix would go undetected. Thus, the burden of detecting false announcements falls on edge networks. If the edge network is monitoring MDP messages, it can quickly detect the false announcement and take action. If the edge network is not monitoring MDP messages, the situation is no worse than it is today. In the current Internet, edge prefixes are announced in BGP. BGP is a path-vector routing protocol, which does not propagate every announcement everywhere. If a prefix is hijacked, the real owner of the prefix may not receive the false announcement, and the attack will go undetected.

A serious attack that a rogue network can launch is to map a large number of edge prefixes to a single ETR. This would redirect a large amount of traffic to that ETR, effectively constituting a distributed denial-of-service (DDoS) attack. To prevent this, DMs sign and announce the list of their own transit prefixes in MDP, propagating the message to every transit network. Receivers can verify the signature and record the list of transit prefixes. To understand how this prevents the aforementioned type of DDoS attack, assume X announces the transit prefix containing ETR e, which is verified and accepted by all other transit networks. If rogue AS Z attempts to map edge prefixes $a/8$ and $b/8$ to e, other transit networks can detect that Z does not own the transit prefix containing e, and will reject the false mapping information.

If Z tries to defeat this scheme by signing and announcing one of X's prefixes in MDP, it will be quickly detected by X. Other networks will detect this conflict as well. They can use past history to help decide which announcement to trust before the problem is resolved. If a network has trusted X's announcement for a long time in the past, it can continue to trust X until the conflict is resolved, likely due to actions X will take.

5. EVALUATION

In this section, we present an evaluation of APT's feasibility using real traffic traces. Whether APT is feasible depends on its data delivery performance and hardware requirements, which in turn are affected by traffic characteristics, since APT uses a data-driven approach to pull mapping information from DMs. We therefore used data-driven simulation to evaluate the packet delay introduced by caching at ITRs, the cache size at ITRs, and the amount of data traffic redirected to DMs. Below, we first describe our simulator and data sources, then present our results.

[3]Note that the monitor does not make any announcements, it simply passively examines all incoming MDP messages.

5.1 The TR Cache Simulator

The cache hit rate at ITRs is critical to overall APT performance. A high hit rate will ensure that few packets will experience redirection delay and each default mapper can serve multiple TRs without being overburdened. To evaluate the TR cache hit rate, and therefore the load placed on default mappers, we simulated TR caching using traces from real provider-edge (PE) routers. We used a number of different cache and network parameters to determine their effect on the cache hit rate.

Our cache simulator examines destination address d of each packet in a traffic trace and attempts to perform a longest-prefix-match lookup of d in its prefix cache, C. If a match is found, this is counted as a cache hit. If no match is found, this is counted as a cache miss and a new cache entry is added for d after a certain delay. The delay is a configurable parameter used to emulate the round-trip time between the ITR and a DM. The prefix used for the new cache entry is determined by a real BGP routing table. This is feasible only when the address d is not anonymized. Otherwise, the simulator uses $d/24$ as the prefix. Note that we are underestimating our cache performance in the latter case, as most prefixes in the BGP routing table are shorter than /24. In reality, we could use a smaller cache and have a lower miss rate.

A maximum cache size m can also be specified. If there is a cache miss when C already contains m entries, the least-recently used prefix is removed from C before the new cache entry is added. Prefixes can optionally be removed from C once they have remained inactive for a specified interval of time, or cache inactivity timeout (CIT).

5.2 Data Sources

We ran the simulator on packet-level traces from two real PE routers.

FRG. This trace was collected at the FrontRange Gigapop in Colorado. It consists of all traffic outbound to a tier-1 ISP during the period 09:00 to 21:00, Mountain Standard Time, on November 7, 2007. In our analysis, we used a list of actual prefixes retrieved from the RIBs at RouteViews Oregon, also on November 7, 2007. When using a limited-size cache with this data set, the maximum size was 4,096 entries, less than ten percent of

Data Source	% Miss Rate					
FRG	0.001	0.002	0.004	0.005	0.537	0.687
CERNET	0.054	0.059	0.198	0.207	0.756	0.810
Delay (ms)	0	50	0	50	0	50
Type	Optimal		With CIT		With Limit	

Table 1: Cumulative cache miss rates for both data sets with three different cache types and best- and worst-case default-mapper latencies.

the total number of prefixes seen in the trace (52,502).

CERNET. This trace was collected at Tsinghua University in Beijing, China. It consists of all traffic outbound from the university through a particular PE router into the CERNET backbone from 09:00 to 21:00, China Standard Time, on January 23, 2008. This data was anonymized using a prefix-preserving method before analysis, so, though addresses remain in the same prefix after anonymization, they cannot be mapped to a real BGP prefix list. Instead, every prefix is assumed to be a /24. This provides us with a worst-case estimate, assuming /24 continues to be the longest prefix length allowed in the network. Since this results in a significantly larger number of total prefixes in the trace (985,757), we used a larger maximum when simulating a limited cache size: 65,536.

5.3 Results

In our simulations, we used four different combinations of cache size and CIT value. The cache size was either unlimited or an order of magnitude smaller than the total number of prefixes seen in the trace. The CIT value was either infinity or 30 minutes. During each run, the simulator emulated four different latencies for retrieving mapping information from a default mapper: zero (an instantaneous cache add), 10ms, 30ms, and 50ms. We selected 50ms as our worst-case delay based on [1] and [13], which show that a single, carefully placed default mapper in the network of most tier-1 ISPs in the United States would be reachable from any hypothetical TR in that network within approximately 50ms.

Table 1 shows cumulative cache miss rates. "Optimal" refers to a cache with unlimited size and an infinite CIT. "With CIT" refers to a cache with unlimited size and a CIT of 30 minutes. "With Limit" refers to a cache with limited size and a CIT of infinity or 30 minutes – the results are the same regardless of the CIT value. This suggests that entries are replaced before their CIT timer expires. Only the best and worst case delays (zero and 50 ms) are shown.

We can make the following two observations. First, the miss rate is well below 1% in all cases. In other words, less than 1% of the traffic was redirected to

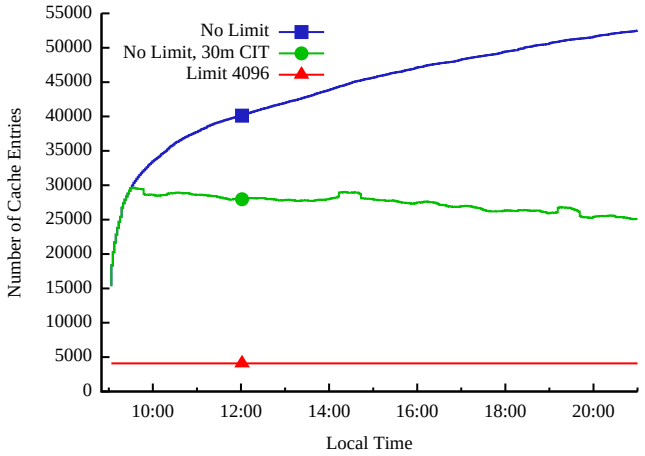

Figure 6: ITR Cache Size (FRG). The first data point was sampled two minutes into the trace.

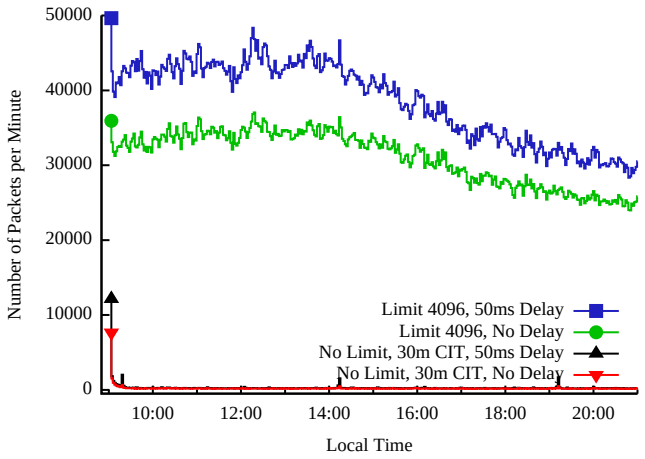

Figure 7: Default Mapper Load (FRG). The first data point was sampled two minutes into the trace.

the local DM. The worst case miss rate is 0.810% for the CERNET data set with a fixed cache-size limit and 50ms delay to receive new mappings. As stated in Section 5.2, we predicted this data set to be a worst case based on our use of /24 prefixes for all addresses.

Second, a 50 ms delay in adding new cache entries had a mostly negligible effect on the miss rate, compared with no delay. One possible explanation is that the inter-packet delay for initial packets to the same destination prefix is longer than 50 ms most of the time (we still need to verify this conjecture).

These results suggests that moving the mapping table from the ITRs to a local DM has negligible impact on overall performance, providing strong support for our design decisions.

Figure 6 shows cache sizes in number of entries and Figure 7 shows the number of packets that would be forwarded to a default mapper per minute, both for the FRG data set. We omit the figures for CERNET, as they are similar to those for FRG.

Two things are apparent from these results. First of all, latency between TR and default mapper has a minimal or, in most cases, undetectable effect on the default mapper load. This is consistent with our earlier results on cache miss rate.

Second of all, the packet-forwarding burden placed on default mappers is quite manageable. Even a TR at a high-traffic, provider-edge router would place a load on the default mapper of less than 1,000 packets per minute in the normal case with a cache size above 30,000 entries. In a more extreme case where such a TR had only a 4,096-entry capacity, the load placed on the default mapper would still be under 50,000 packets per minute. Using this data, we can make a conservative estimate of the number of TRs that a single default mapper can support. Assuming the worst case from our simulations of 50,000 redirected packets per minute per TR, even a default mapper running on commodity 2001 PC hardware would have enough forwarding capability to support hundreds of TRs [19].

6. INCREMENTAL DEPLOYMENT

On the Internet, one simply cannot set a flag day when all sites will switch to a new design, no matter how great an advantage the design offers. As a result, APT explicitly assumes incremental deployment. Our design offers incentives for sites that adopt APT. An APT-capable ISP will be able to reduce the routing table size in its internal routers. Moreover, our design allows backwards compatibility for sites that are slow to adopt APT by converting mapping information in APT networks to BGP routes that can be used by legacy networks.

Before we delve into the details, we define the following terms. If a transit AS has adopted APT, it is called an *APT AS*. Otherwise, it is called a *non-APT AS*. A topologically connected set of APT ASes form an *APT island*. Note that our design allows individual ISPs to deploy APT unilaterally, without any coordination with other ASes. Such an ISP would simply form a new APT island. Unconnected APT islands do not exchange mapping information with each other.

6.1 Edge Networks

APT offers various incentives for edge networks to use APT providers. The Map N Encap solution allows all edge networks to use provider-independent addressing, which eliminates forced renumbering due to provider change, and also eases multihoming. In addition, APT mappings are a powerful tool for traffic engineering. Currently, an edge network may use AS-path padding or address de-aggregation for load balancing. However, these techniques provide only rudimentary control over which route is selected by a traffic source. In APT, an edge network can clearly specify traffic preferences among all of its APT providers. This explicit approach to managing inbound traffic greatly simplifies existing practices and achieves more effective results.

These benefits come at minimal to no cost for edge networks. Because the APT design focuses on placing new functionality in transit networks, all changes go virtually unnoticed by edge networks. The only new task for an edge network is to provide traffic preference information to its providers. If necessary, a transit provider can generate this traffic engineering information on behalf of its edge-network customers, and APT can be incrementally deployed *without any changes* to edge networks.

6.2 Transit Networks

All transit ASes will continue to use BGP to reach transit prefixes, even if all of them adopt APT. Edge prefixes are handled differently. APT islands configure their border routers as TRs so that their customers' data packets will be encapsulated and decapsulated as they enter and exit the AS. An APT island can then remove all customer edge prefixes from their BGP routing tables.

APT ASes must still allow their customers to interact with the rest of the existing system. To explain how this is done, we must answer three questions:

What information do APT ASes use to reach their customer edge prefixes? Inside an APT island, the APT ASes exchange mapping information with each other (see Section 4.5). This allows their default mappers to maintain a mapping information table for the entire island. We will call this the *island mapping table*.

How can an APT AS reach edge prefixes served by non-APT ASes? All transit ASes will continue to use BGP to reach those edge prefixes connected to non-

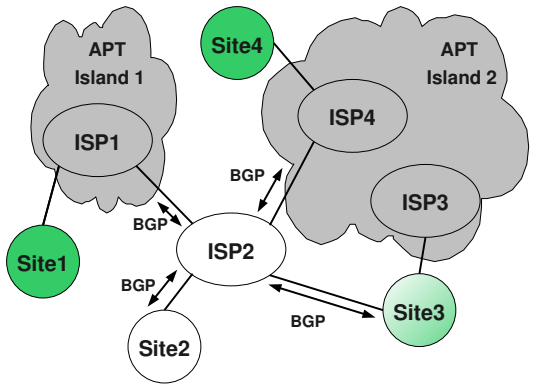

Figure 8: **Example Topology for Incremental Deployment**

APT ASes. Note the following differences from the current Internet: (a) APT ASes do not run BGP sessions with their customer networks in edge address space, and (b) the BGP routing tables maintained by routers in APT ASes do not contain those edge prefixes that are already in the island mapping table (unless a prefix is connected to both an APT AS and a non-APT AS. See section 6.3).

How can an edge network connected to a non-APT AS reach an edge prefix connected to an APT AS? APT ASes at the border of an APT island must advertise the edge prefixes in their island mapping table to their non-APT neighbors via BGP.

An APT island grows larger by merging with another APT island. When two APT islands merge, their island mapping tables merge into a single, larger island mapping table. As a result, each router in the merged island can remove the island mapping table prefixes from their BGP tables, offsetting the increase in mapping table size. Furthermore, the increase in mapping table size will affect only a small set of devices (default mappers), while essentially all routers can benefit from the reduction in BGP table size. As the APT island grows, the BGP tables of the island routers will continue to shrink, providing incentive for non-APT ASes to join the island (and for APT islands to merge). APT providers can also offer their customers all of the benefits mentioned in Section 6.1.

6.3 Interoperation Under Partial Deployment

We now describe how to enable the communication between APT and non-APT networks, or between two different islands, using the topology in Figure 8. Suppose edge network *Site1* is a customer of *ISP1*, and thus is a part of *APT Island 1*. *Site3* and *Site4* are customers of *ISP3* and *ISP4* respectively. They are part of *APT Island 2*. *Site2* is a customer of *ISP2*, which is a non-APT network. *Site3* is also a customer of *ISP2*.

How can a non-APT site like *Site2* reach an APT site, such as *Site1*? Recall that *Site1*'s prefixes are not in the BGP tables of any router in *APT Island 1*, but they *are* in the *APT Island 1* mapping table. Thus, ISPs at the border of Island 1 need to convert the mapping information for *Site1* into a BGP route and inject it into non-APT networks. Since default mappers maintain a complete island mapping table, they can do the conversion – the converted BGP route will contain only the announcing DM's own AS number (the AS where traffic will enter the island) and *ISP1* (the AS where traffic will exit the island towards *Site1*). In addition, if *Site1* has an AS number, its AS number will appear at the end of the BGP path in order to be consistent with current BGP path semantics. The details of the path taken within the APT island are not relevant to the BGP routers in the legacy system. DMs will advertise these routes to their networks' non-APT neighbors in accordance with routing policies. Eventually, *Site2* will receive the BGP route to *Site1*. These APT BGP announcements will include a unique community tag X so that other BGP speakers in *APT Island 1* can ignore them.

The above works fine for sites whose providers are all from the same APT island, but what about sites that multihome with ISPs both inside and outside of the island? To support this type of multihoming, we require that all APT routers check their BGP tables *before* attempting to encapsulate a packet. Otherwise, packets would always route through APT providers to the destination site, never using the non-APT provider. Furthermore, the DMs at island border ISPs will still announce these sites' prefixes into BGP, but will tag these announcements with a unique community tag Y(different from X) telling other BGP speakers in the island that the destination sites are multihomed to ASes inside and outside the island. Note that Y must differ from X. BGP announcements with community tag X can be ignored by non-DM routers in the APT Island. However, announcements with community tag Y cannot be ignored by island nodes.[4]

To see how these requirements support the above multihoming, we will go through an example. In Figure 8, *Site3* multihomes with an APT AS(*ISP3*) as well as a non-APT AS(*ISP2*). Thus *Site3* will have 2 types of routes announced into BGP – a traditional BGP route announced by *ISP2*, and an injected BGP route announced by APT ISPs at the border of *APT Island 2*. The injected BGP route will include a unique community tag Y telling other BGP speakers in *APT Island 2* that *Site3* is multihomed to ASes inside and outside *APT Island 2*. Receivers of the announcements

[4]More specifically, the announcements cannot be ignored by ITRs and island border routers that peer with non-island neighbors. Other island routers can still ignore the announcements.

will choose one route to store in their loc-RIB, using standard BGP route selection. When a border router in *APT Island 2* receives packets destined to *Site3*, it first checks its BGP table *before* looking in its cache. It will find one of the 2 BGP routes in its loc-RIB. It then checks the route community attribute value. If the value is Y, then it knows the route is an injected route, and it attempts to encapsulate the packet via standard APT practices. If the value is anything other than Y, the router does not encapsulate the packet and routes the packet via standard BGP.

We now explain how an APT site can communicate with an non-APT site. For example, how can *Site1* reach *Site2*? When an ITR in *ISP1* receives a packet from *Site1*, it first looks for the prefix in its BGP routing table (as mentioned in the previous example). Since non-APT prefixes are stored in a TR's BGP routing table, the ITR will find a match, check the route's community attribute, and discover that the prefix belongs to a non-APT AS. The packet is then forwarded toward the destination using the forwarding table generated by BGP.

How do two unconnected APT islands communicate with each other? In our figure, *Site4* is a customer of *ISP4*, an APT network, but *ISP4* is not in the same island as *Site1*'s provider, *ISP1* (i.e. there are some non-APT networks in between). Unconnected APT islands do not exchange mapping information with each other, so *Site4*'s prefixes will not be in *APT Island 1*'s mapping table, and *Site1*'s prefixes will not be in *APT Island 2*'s mapping table. However, the two islands will still receive each other's BGP routes injected using the method described previously. As a result, *Site1* will communicate with *Site4* just as it would with the customer of a non-APT network, and vice versa.

7. ROUTING POLICY AND MAPPING

As previously noted, the inter-domain routing protocol is outside the scope of the APT design. If APT were deployed on the current Internet, BGP would continue to serve this purpose. In other words, BGP will still be used to find paths between ITRs and ETRs that are in different ASes.

However, an ETR is a necessary hop in any APT routing path, but multihomed destinations have more than one ETR to choose from. Therefore, APT ETR selection can have an effect on routing paths. In this section, we intend to clarify how APT can affect BGP routing paths, and what kinds of policies are both possible and necessary to support in APT to maintain the flexibility of current routing policy.

One might believe that there are three situations in which policy can be applied to mapping information in APT: (1) When a provider-specific MapSet is created, (2) when a default mapper selects an ETR from a MapSet, and (3) when propagating MapSets to other transit networks. However, APT chooses to make policy applied in situation 1 take first priority and use situation 2 only to break ties. We believe that source-specific mappings are too expensive to support; they would defeat our hybrid push-pull approach. Therefore, APT negates the usefulness of situation 3.

To understand why, consider the following. Since the path taken by a BGP update determines the path of data flow, the path of each BGP update must be carefully managed through policy. This is not the case for MapSet announcements. MapSets do not change based on the path by which they are propagated. In fact, APT guarantees this – any modification made to a MapSet during propagation will cause signature verification to fail and propagation to end. Furthermore, it is in the interest of the party owning an ITR, or sending party, to have access to *all* MapSets in the network. This will allow the sending party to provide the most robust service to their customers.

The result is that applying policy along the path via which a MapSet is propagated will not have any desirable effect. For example, assume, for the sake of argument, we used a policy-rich protocol, such as BGP, for MapSet update propagation. Accordingly, some transit network X withholds an update for some MapSet m from their peer Y. Y wants to receive all updates for all MapSets, so Y simply peers with Z, who *is* willing to send updates for m. The MapSet updates for m that Y receives from Z are *identical* to the updates that it would have received from X, were X willing to forward them. Therefore, all that X has accomplished by withholding MapSet updates from Y is to force Y to find an additional peer. More importantly, X's application of policy has not had any effect on the routing paths between X and Y. This is due to the fact that the method by which Y selects an ETR for any given destination edge address is entirely unrelated to the method by which it received the corresponding MapSet.

8. RELATED WORK

Network routing is a very active and fruitful research area. We only mention a sample set of related work here.

Several research efforts took a clean-slate approach to new routing architecture design. One recent effort, named Cabo [8], divides the Internet into 2 groups of players, "Service Providers" and "Infrastructure Providers". Service providers buy resources from infrastructure providers in order to provide services to Internet users. Cabo focuses on enabling new end-to-end services that users can choose from, rather than the routing scalability problem. Another clean-slate approach, NIRA [24], explores the use of source routing to allow end users to choose from different ISP paths. Another research project,

MIRO [23], also promotes user choices. MIRO allows users to select alternative AS paths (other than the default BGP route) in order to satisfy desired end-to-end path properties. Again, routing scalability was not the primary goal of this effort.

Subramanian et al. proposed HLP [22] to address the routing scalability problem. HLP divides the Internet routing infrastructure into many trees, each with tier-1 providers as the root. The design goal is to confine local routing instability and faults to each tree. However, as noted by the HLP designers, Internet AS connectivity does not match well to a model of non-overlapping trees. In fact, multihoming practices have been increasing rapidly over time, which stands in direct opposition to HLP's attempt to divide the routing infrastructure into separable trees. In contrast, APT separates the transit core of the routing infrastructure from the edge networks, greatly facilitating edge multihoming.

CRIO [26] represents another effort to address routing scalability. To reduce the global routing table size, CRIO proposes to aggregate otherwise non-aggregatable edge prefixes into "virtual prefixes". The routers that advertise these virtual prefixes become the proxy tunnel ends for traffic going to the prefixes they aggregate. Thus, some traffic may take a longer path.

On the operational Internet, the inherent conflict between provider-based addressing and site multihoming has long been recognized. Two solutions to the problem, Map & Encap [4, 10] and GSE [20] were proposed more than ten years ago. Both proposals separate edge networks from the transit core in the routing system. GSE uses the low-order bytes of IPv6 addresses to represent the address space inside edge networks, and the high-order bytes for routing in the transit core. Like Map & Encap, GSE needs a mapping service to bind the two address spaces. They propose storing the mapping information in DNS. This approach avoids the need for a mapping system such as APT, but brings up a number of other issues. [25] provides an overview of open issues with GSE, some of which are shared by any routing separation design, *e.g.*, handling border link failures and edge-network traffic engineering, which are addressed in APT.

Since 2007, the IRTF Routing Research Group has been actively exploring the design space for a scalable Internet routing architecture. Among the proposed solutions, a notable one is LISP [7] and its associated mapping services, CONS [6] and ALT [5]. Collectively, they represent another realization of the Map & Encap scheme, which differs in a number of significant ways from APT. One difference is in mapping information distribution. APT distributes a full mapping table to every transit AS, allowing each AS to decide how many DMs to deploy to balance the tradeoff of cost versus performance. CONS and ALT keep the mapping information at the originating edge networks, and build a global hierarchy of servers to forward mapping requests and replies. Another major difference is the location of TRs: APT prefers provider-edge routers to align cost with benefit as well as facilitate incremental deployment, while LISP prefers TR deployment at customer-edge routers.

[12] reported the results of an evaluation of ITR caching performance in LISP using traffic traces collected between a university campus and its ISP. It demonstrated the effects of cache size, lifetime, and cache miss rate, and the impact on traffic. We also evaluated APT performance using data traces collected from operational networks. While [12] uses data from one edge network (which is appropriate for LISP), our evaluation is based on data traces from provider-edge routers that typically serve multiple edge-network customers.

Another approach to reduce routing table size is to use compact routing, *i.e.*, trade longer paths for less routing state. However, a recent study determined that this type of routing cannot handle routing dynamics very well. [14]

9. CONCLUSION

In this paper, we have presented a practical design for a new tunneling architecture to solve the routing scalability problem. To summarize our design, APT deploys default mappers in transit networks to maintain the full table of mappings from edge prefixes to the addresses of their transit providers, so that data packets can be tunneled over the transit core. The DMs form a mesh congruent to the underlying network topology and use the mesh to flood mapping information. To secure mapping data distribution and all control messages, DMs cryptographically sign messages and use a novel scheme based on neighbor signatures to distribute public keys. To minimize control overhead, data delay, and data loss, APT adopts a data-driven approach to handle cache misses at ITRs as well as temporary unreachability of ETRs; data packets are used both to signal DMs to provide mapping information to ITRs and to allow DMs to forward these data packets in the meantime.

Looking at the bigger picture, APT necessarily brings additional complexity into the Internet architecture. Thus, a question naturally arises: why is it necessary to change the existing routing architecture?

We believe the answer lies in the fact that the Internet has grown by orders of magnitude. In a 1928 article by J. B. S. Haldane, "Being the right size" [9], the author illustrated the relationship between the size and complexity of biological entities using a vivid example. As stated in the article, "a typical small animal, say a microscopic worm or rotifer, has a smooth skin through which all the oxygen it requires can soak in." However,

"increase its dimensions tenfold in every direction, and its weight is increased a thousand times, so ... it will need a thousand times as much food and oxygen per day. Now if its shape is unaltered its surface will be increased only a hundredfold, and ten times as much oxygen must enter per minute through each square millimeter of skin." This is why every large animal has a lung, an organ specialized for soaking up oxygen. The author concludes that, "for every type of animal there is a most convenient size, and a large change in size inevitably carries with it a change of form." It would be unimaginable for small insects to have lungs. On the other hand, it is also impossible for big animals to live without lungs.

In the case of the Internet, the existing architecture, where all autonomous systems live in the same routing space, was designed more than a decade ago when the Internet was very small in size. Today, not only has the Internet grown beyond its designers' wildest imaginations, but the goals of individual networks have diverged. Edge sites are multihomed for enhanced reliability and performance, while ISPs are specialized for high-performance, yet economical, packet delivery service. The different goals of different parties have brought different and conflicting requirements to the shared address and routing space. Thus, the original architecture can no longer meet the functional requirements of today's grown-up Internet. A new routing architecture is needed to accommodate the growth of the Internet and the differentiation of individual networks, and APT is exactly such an attempt.

10. ADDITIONAL AUTHORS

11. REFERENCES

[1] ATT. ATT US network latency. `http://ipnetwork.bgtmo.ip.att.net/pws/network_delay.html`.

[2] T. Bu, L. Gao, and D. Towsley. On characterizing BGP routing table growth. *Computer Networks*, 45(1):45–54, May 2004.

[3] M. Caesar, T. Condie, J. Kannan, K. Lakshminarayanan, I. Stoica, and S. Shenker. ROFL: Routing on Flat Labels. In *Proc. of the ACM SIGCOMM*, 2006.

[4] S. Deering. The Map & Encap Scheme for Scalable IPv4 Routing with Portable Site Prefixes. Presentation, Xerox PARC, March 1996.

[5] D. Farinacci, V. Fuller, and D. Meyer. LISP Alternative Topology (LISP-ALT). draft-fuller-lisp-alt-01.txt, 2007.

[6] D. Farinacci, V. Fuller, and D. Meyer. LISP-CONS: A Content distribution Overlay Network Service for LISP. draft-fuller-lisp-cons-03.txt, 2007.

[7] D. Farinacci, V. Fuller, D. Oran, and D. Meyer. Locator/ID Separation Protocol (LISP). draft-farinacci-lisp-05.txt, 2007.

[8] N. Feamster, L. Gao, and J. Rexford. How to lease the Internet in your spare time. *ACM SIGCOMM CCR*, 37(1):61–64, 2007.

[9] J. B. S. Haldane. Being the Right Size. `http://irl.cs.ucla.edu/papers/right-size.html`, 1928.

[10] R. Hinden. New Scheme for Internet Routing and Addressing (ENCAPS) for IPNG. *RFC 1955*, 1996.

[11] G. Huston. Analyzing the Internet BGP routing table. *Internet Protocol Journal*, 4(1), 2001.

[12] L. Iannone and O. Bonaventure. On the cost of caching locator/ID mappings. In *Proc. of the CoNext Conference*, 2007.

[13] Keynote. Internet health report. `http://www.internethealthreport.com/`.

[14] D. Krioukov, kc claffy, K. Fall, and A. Brady. On compact routing for the Internet. *ACM SIGCOMM CCR*, 37(3):43–52, July 2007.

[15] J. Li, M. Guidero, Z. Wu, E. Purpus, and T. Ehrenkranz. BGP routing dynamics revisited. *ACM SIGCOMM CCR*, 37(2):7–16, Apr. 2007.

[16] D. Massey, L. Wang, B. Zhang, and L. Zhang. A scalable routing system design for future Internet. In *Proc. of the ACM SIGCOMM Workshop on IPv6 and the Future of the Internet*, Aug. 2007.

[17] X. Meng, Z. Xu, B. Zhang, G. Huston, S. Lu, and L. Zhang. IPv4 Address Allocation and BGP Routing Table Evolution. In *ACM SIGCOMM CCR*, Janurary 2005.

[18] D. Meyer, L. Zhang, and K. Fall. Report from the IAB Workshop on Routing and Addressing. draft-iab-raws-report-01.txt, 2007.

[19] R. Morris, E. Kohler, J. Jannotti, and M. F. Kaashoek. The click modular router. *SIGOPS Oper. Syst. Rev.*, 33(5):217–231, 1999.

[20] M. O'Dell. GSE - An Alternate Addressing Architecture for IPv6. February 1997.

[21] R. Oliveira, R. Izhak-Ratzin, B. Zhang, and L. Zhang. Measurement of Highly Active Prefixes in BGP. In *IEEE GLOBECOM*, 2005.

[22] L. Subramanian, M. Caesar, C. T. Ee, M. Handley, Z. M. Mao, S. Shenker, and I. Stoica. HLP: A Next Generation Inter–domain Routing Protocol. In *ACM SIGCOMM*, 2005.

[23] W. Xu and J. Rexford. MIRO: Multi-Path Interdomain Routing. In *Proc. of the ACM SIGCOMM*, 2006.

[24] X. Yang, D. Clark, and A. Berger. NIRA: A new routing architecture. *IEEE/ACM Transactions on Networking*, 15(4), Aug. 2007.

[25] L. Zhang. An overview of multihoming and open issues in GSE. *IETF Journal*, 2, 2006.

[26] X. Zhang, P. Francis, J. Wang, and K. Yoshida. Scaling IP Routing with the Core Router-Integrated Overlay. In *Proc. of ICNP*, 2006.