Inter-Domain Routing with Cut-Through Switching for the MobilityFirst Future Internet Architecture

Adrian Lara[†] , Shreyasee Mukherjee[††], Byrav Ramamurthy[‡], Dipankar Raychaudhuri[††] and K. K. Ramakrishnan[‡‡]

[†] University of Costa Rica, San Jose, Costa Rica, adrian.lara@ecci.ucr.ac.cr

[††] WINLAB, Rutgers University, New Brunswick, NJ 08901-8541, USA, {shreya, ray}@winlab.rutgers.edu

[‡] University of Nebraska-Lincoln, Lincoln, NE 68588-0115, USA, byrav@cse.un.edu

[‡‡] University of California-Riverside, Riverside, CA 92521-9800, USA, kk@cs.ucr.edu

***Abstract*—Future Internet projects such as MobilityFirst and Named Data Networking have proposed novel mechanisms to replace the Internet Protocol to better support content delivery and mobility. However, the problem of efficient data transfer across the network core has not been adequately investigated. We tackle the challenge of inter-domain cut-through switching using software-defined networking (SDN). First, we propose and solve an optimization problem that minimizes the total transfer time using inter-domain tunnels. Second, we propose an SDN-based routing framework for the MobilityFirst architecture capable of dynamically creating such tunnels. The main novelty of this framework is to name tunnels as network objects to simplify how tunnels are created and maintained. To validate our framework, we implement on the GENI (Global Environment for Network Innovations) testbed a prototype for the MobilityFirst architecture. Our experiments with the optimization problem show that the inter-domain latency between controllers plays a key role on how tunnels are setup. Furthermore, our implementation experiments show that the control plane delay can be reduced by 75% when using inter- domain tunnels. Finally, we show how our framework needs fewer messages than current protocols such as label distribution protocol (LDP) to setup intra- domain and inter-domain tunnels.**

I. Introduction

Future Internet Architectures (FIAs) redesign the narrow waist (IP layer) of the Internet to better support mobility, efficient content delivery and trustworthiness. For example, MobilityFirst [1] embraces several key concepts centered around secure global identifiers that inherently support mobility and trustworthiness. This includes a hop-by-hop segmented data transport based on a globally unique identifier (GUID) and a global name resolution service (GNRS), as well as name/address-based hybrid routing. Similarly, in Named Data Networking (NDN) [2], IP addresses are replaced by content requests and responses to allow users to focus on data rather than sources and destinations.

These FIA networks face a challenge that has not been solved yet. While significant attention has been paid to edge-awareness and in-network caching, very little attention has been focused on efficient transmission of data across the network core. Indeed, an efficient technique to traverse one or many autonomous systems (ASes) is as important as edge-awareness to ensure efficient content delivery. To address this, we propose using cut-through switching tunnels to bypass the routing mechanisms for certain flows that benefit from such fast paths. To realize this, we leverage software-defined networking (SDN) as a key technology to make the network programmable and more flexible.

This material is based upon work supported by the National Science Foundation under Grants No. CNS-1345277, CNS-1345295 and CNS-1455815.

To motivate the need for inter-domain cut-through switching in SDN, we first model the dynamic creation of inter-domain tunnels as a linear optimization problem. Particularly, we describe this problem in the context of inter-domain SDN. The problem minimizes the total transfer time while considering the costs of creating inter-domain tunnels. Using this problem formulation, we demonstrate how inter-domain controller latency plays a key role on how tunnels are created. Indeed, we show how inter-domain tunnels are better when the latency is small, but intra-domain tunnels are better otherwise. To the best of our knowledge, this paper is the first one to formulate and solve the creation of inter-domain tunnels in an SDN-controlled network.

After that, we propose an routing framework for MobilityFirst that enables dynamic inter-domain cut-through switching. The framework is based on the following design requirements: inter-domain topology visibility, naming the tunnels as network objects and per-flow traffic engineering. To demonstrate the feasibility of the proposed framework, we develop a prototype for MobilityFirst using the GENI testbed. The results show that in-transit *packet_in* messages can be reduced by 75% using inter-domain tunnels. Furthermore, naming tunnels as network objects scales better than current protocols such as LDP to setup tunnels.

The remaining of this paper is organized as follows. We first survey the related work in Section II. Next we describe the optimization problem in Section III and we describe the routing framework in Section IV. After that, we evaluate our work in Section V and conclude in Section VI.

II. Background and Related Work

A. Overview of MobilityFirst

The MobilityFirst architecture uses flat names as identifiers and separates the naming of entities (identifiers) from their addressing (location). As shown in Fig. 1, every object attached to the network is assigned a flat 160 bit globally unique identifier (GUID) by one of several name certification services (NCS). These GUIDs serve as self-certifying consistent names for not only individual devices such as 'John's laptop' or 'Sensor XYZ,' but also for groups of devices ('all of Sue's devices'), a context ('taxis in Lincoln') or even more abstract entities such as 'all OpenFlow capable routers in a network' or 'all routers along path ABC.' The mapping of GUIDs to their network attachment points (NAs) is maintained by a dynamic mapping entity that is globally distributed but logically centralized called the global name resolution service (GNRS) [3]. Every time a network entity changes its point of attachment to the Internet, the mapping in the GNRS is updated. Furthermore, routers have the capability of querying the GNRS for up-to-date GUID-NA mappings.

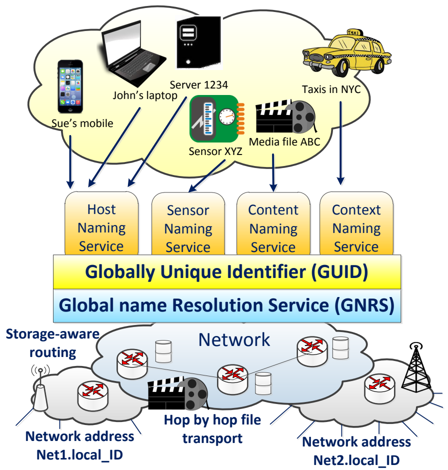


Fig. 1: Overview of the MobilityFirst architecture and its components

B. Intra-domain cut-through switching in MobilityFirst using SDN

To explore how MobilityFirst can benefit from cut-through switching, we implemented a prototype MobilityFirst using an SDN-based control plane [4]. In this work we demonstrated the gain in performance between having an end-to-end tunnel and forwarding on a hop-by-hop basis and the benefits of aggregating multiple flows in a single tunnel at intra-domain scale. The performance gain achieved motivated us to investigate if such benefits are possible at inter-domain scale as well.

C. Multi-domain optimization

Existing studies have considered inter-domain routing as an optimization problem [5], [6], [7]. Tomaszewski et al. [5] consider the problem of bandwidth reservation on inter-domain links for different traffic classes. Roughan et al. [6] tackle the problem of traffic engineering with limited information shared across domain. Finally, Chamania et al. [7] explore how to achieve IP routing stability through dynamic creation of tunnels at the WDM layer. Our model of tunnels, described in the next section, is based on the formulation proposed by this work.

III. Dynamic Creation of Inter-Domain Tunnels

In this section, we model the dynamic creation of inter-domain tunnels as an optimization problem. The objective function minimizes the total transfer time, including control plane delays, while considering the different costs of creating and maintaining inter-domain tunnels.

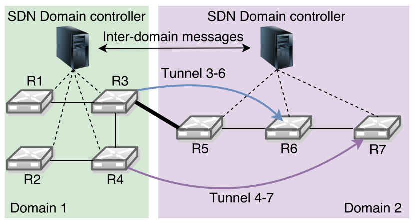


Fig. 2: Sample network with two domains and two cut-through tunnels (3-6 and 4-7).

A. Assumptions

We make the following assumptions in the formulation of this problem. First, we assume that the computation of the tunnels is being performed by the controller of one of the domains. Moreover, we assume that such controller has visibility into the bandwidth of all links, including those belonging to other domains. In the framework described in Section IV we propose a way to achieve this. Second, we assume that, when the optimization begins, no tunnels exist and there is no traffic flowing. We leave studying the steady-state scenario for future work. Third, we assume that the cost of inter-domain tunnels is computed based on the individual cost of each link traversed by the tunnel. Fourth, we assume that the flow rate and duration between each source and destination node are known. Finally, we assume that only one tunnel can exist between two routers, but the tunnel can carry more than one flow between different sources and destinations.

B. Settings

Consider a network of V nodes (OpenFlow-compliant routers) belonging to at least two different domains. The path between all pairs of nodes is known and the optimization problem explores all combinations of using links individually or in tunnels.

We consider the following parameters:

- $\lambda_{s,d}$: Bit rate in Mbps between source s and destination $v \in V$;
- $\delta_{s,d}$: Duration in ms of flow between source s and destination $v \in V$;
- $c_{i,j}$: Capacity in Mbps of link i, j;
- $\psi^{i,j}_{x,y}$: Link i, $j \in L$ is used when a tunnel between x, $y \in L$ is setup (boolean);
- $\phi^{i,j}_{s,d}$: Link i, $j \in L$ belongs to the path between s and $d \in V$ (boolean);

- $b_{i,j}$: Time in ms needed to setup inter-domain tunnel when link i, j is used;
- $m_{i,j}$: Maintenance cost due to inter-domain messages to maintain a tunnel when link i, j is part of it;
- l_x: Latency in ms between switch $x \in V$ and the controller it is connected to;
- s_x: Time in ms needed by the controller to handle a packet_in message sent by switch $x \in V$;
- u_x: Time in ms needed by the initiating domain controller to compute a tunnel path;
- $w_{x,y}$: Inter-domain latency in ms between domain controllers to setup a tunnel between switches x and $y \in V$;
- T: Maximum number of tunnels allowed;
- M: Maximum maintenance cost allowed per domain;
- D: A flow duration must be at least D times longer than the time needed to create a tunnel in order to be routed through that tunnel. This parameter is used in constraint 6 and is explained below in more detail.

Likewise, we consider the following decision variables:

- $t_{x,y}$: A tunnel between nodes $x, y \in V$ is created.
- $f^{s,d}_{x,y}$: Flow between $s, d \in V$ is routed through a tunnel between nodes $x, y \in V$.
- $r^{i,j}_{s,d}$: Flow between $s, d \in V$ is routed through direct link i, j.

We also use use two additional parameters to simplify the objective function:

- $\tau_{x,y}$: Time needed to setup a tunnel between switches x and $y \in V$. This parameter is computed as $\tau_{x,y} = \psi^{i,j}_{x,y} \times l_x + u_x + w_{x,y}$ (i.e. the sum of the switch to link latencies, the inter-domain controller latency and the time needed by the initiating controller to calculate the path).
- $\iota_{i,j}$: Time needed by the domain controller to handle a packet_in message when a flow is routed through the direct link $i, j \in V$. This parameter is computed as $\iota_{i,j} = 2 \times l_j + s_j$ (i.e. the round-trip latency between the switch and the controller added to the time needed by the controller to handle the packet_in message).

C. Problem formulation

For each flow, there is a known transfer time $\delta_{s,d}$. However, additional delays happen every time a *packet_in* message is received a rule is inserted in the flow table of the switch. Similarly, the flow can be delayed if a cut-through tunnel is being created. The goal of this problem is to minimize the total transfer time for all flows combined with the delays caused by the control plane.

Objective function: Minimize

$$\sum_{s,d \in V} \delta_{s,d} + \sum_{x,y,i,j \in V} r^{i,j}_{s,d} \times \iota_{i,j} + f^{s,d}_{x,y} \times \tau_{x,y} \tag{1}$$

Subject to:

$$\forall i, j \in V, \sum_{x,y,s,d \in V} \lambda \times (\phi^{i,j}_{s,d} \times r^{i,j}_{s,d} + \psi^{i,j}_{x,y} \times f^{s,d}_{x,y}) \leq c_{i,j} \tag{2}$$

$$\forall i, j, s, d \in V, \phi^{i,j}_{s,d} \times r^{i,j}_{s,d} + \sum_{x,y \in V} \psi^{i,j}_{x,y} \times f^{s,d}_{x,y} = 1 \tag{3}$$

$$\forall x, y, s, d \in V, t_{x,y} \geq f^{s,d}_{x,y} \tag{4}$$

$$\forall x, y \in V, \sum t_{x,y} \leq T \tag{5}$$

$$\forall s, d \in V, D \times \sum_{x,y \in V} f^{s,d}_{x,y} \times \tau_{x,y} \leq \delta_{s,d} \tag{6}$$

$$\forall x, y \in V, \sum t_{x,y} (\sum_{i,j \in V} m_{i,j} \times \psi^{i,j}_{x,y}) \leq M \tag{7}$$

The minimization objective function of this problem (Equation 1) uses the τ and ι parameters to take into account all the delays. For each direct link used, the ι delay is considered. Similarly, for each tunnel, the τ delay is counted. By adding these delays to the duration of each flow $\delta_{s,d}$, the problem minimizes the total transfer time.

Constraint 2 ensures that the link capacity limit is enforced for all links in the network. Constraint 3 ensures that all the links $\{i, j\}$ between a source s and a destination d are used, either as a single link ($\phi^{i,j}_{s,d} \times r^{i,j}_{s,d}$), or as part of a tunnel that uses that link ($\psi^{i,j}_{x,y} \times f^{s,d}_{x,y}$). By making the equation equal to one, we also guarantee that only zero or one bypass using the link can co-exist, thus eliminating incompatible tunnels.

Constraint 4 is used to ensure that $t_{x,y}$ is true if, for any pair of source s and destination d, the tunnel $f^{s,d}_{x,y}$ is used at least once. $t_{x,y}$ is then used to keep track of the maintenance cost of the system, instead of using f values that are specific to each flow and could thus be duplicated for the same pair of nodes.

Constraint 5 guarantees that the number of tunnels created stays below T, the maximum number of tunnels allowed.

Finally, Constraint 6 ensures that a flow between source s and destination $d \in V$ only goes through tunnels if the time needed to setup the tunnels is D times smaller. The goal of this constraint is to avoid tunneling a flow that lasts 5 seconds if creating a tunnel will take 3 seconds. Instead, a longer flow will benefit more from the tunnel. Similarly, constraint 7 ensures that the maintenance cost of all tunnels is below the maximum threshold M.

D. Study of sample topologies

To validate the problem formulation, we ran the optimization for two different topologies created randomly. Topology 1 is a small topology with 11 nodes, two domains and 6 flow requests. Topology 2 has 35 nodes, two domains and 15 flow requests. In our experiments, all flow requests are inter-domain.

First, we note that the inter-domain latency plays a key role in how tunnels are created. We experimented using Topology 1 for different values of inter-domain controller latency across domains (w parameter in the optimization problem). The results in Fig. 3a show how the number of inter-domain tunnels created decreases as the latency increases. Indeed, the benefit of bypassing multiple hops (i.e. reducing the impact of switch to controller latency at each hop) is only beneficial as long as the time needed to create a tunnel is reasonable. As a result, for a small inter-domain latency, the optimization problem tends to create a single tunnel for each flow. However, when the

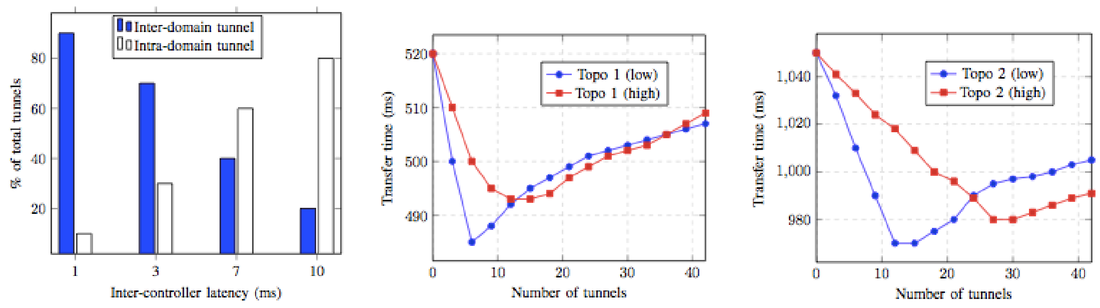


Fig. 3: **(a)** The distribution of intra-domain and inter-domain tunnels for varying inter-domain controller latency. **(b)** and **(c)** The total delay caused by controller processing with low and high inter-controller latency for Topology 1.

value increases, the solver tends to create more edge-to-edge intra-domain tunnels for each domain.

Such a behavior is confirmed in Fig. 3b and Fig. 3c. We made a modification to Constraint 5 to enforce an exact number of tunnels instead of just setting an upper bound T. As a result, we observe the minimum delays obtained for topologies 1 and 2, when the inter-domain controller latency is 3 or 10 (low and high in Fig. 3a). Notice how, for a low latency, the solver creates approximately one tunnel per flow. In contrast, for a higher latency, on average two tunnels per flow are created.

Finally, these experiments also show that the maximum delay is obtained when there are no tunnels. Likewise, once the minimum has been obtained, the increase in the delay is slow in comparison to how it decreases before reaching the optimal value. Therefore, we can conclude that the best way to set up the tunnels depends on the inter-domain controller latency and also having too many tunnels is better than not having enough tunnels.

In a real network, several implementation challenges must be addressed to solve the problem of inter-domain cut-through switching. For example, we assumed a known data rate and flow duration, as well as inter-domain link visibility. After demonstrating the importance of inter-domain controller latency in this section, next we propose a routing framework capable of inter-domain cut-through switching that simplifies how the tunnels are created and maintained.

IV. Routing Framework Description

To implement dynamic creation of inter-domain tunnels, three requirements must be met by the routing framework. First, domain controllers need to be aware of at least part of the topology from other domains (we assumed in the formulation problem that each domain controller was aware of the bandwidth of all links in the network). Second, the domain controllers must be capable of exchanging messages with other domains to setup cut-through tunnels across domains. Third, the controllers must be capable of traffic engineering to decide how to map flows to tunnels.

The description of this framework uses MobilityFirst terminology. However, the proposed work can function in any name-based network with a centralized global name resolution mechanism.

A. Increased visibility between domains

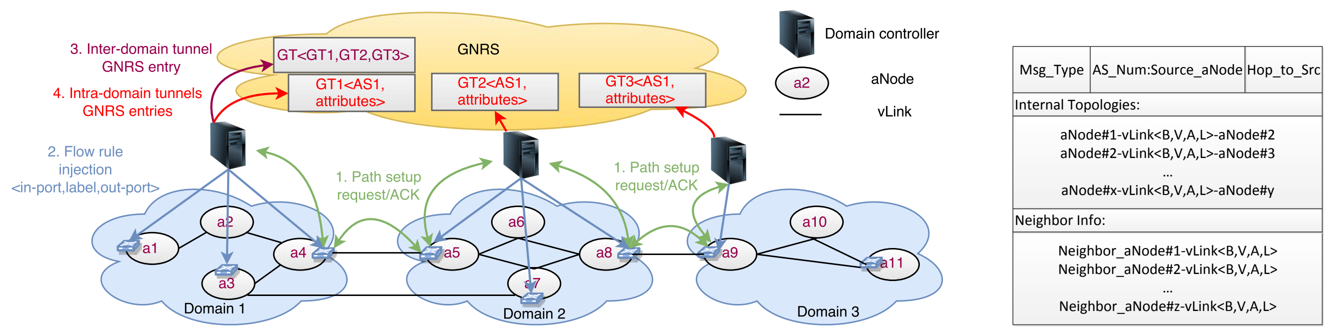
In MobilityFirst, autonomous systems (ASes) have the flexibility to expose their internal network characteristics in terms of aggregated nodes (aNodes) and virtual links (vLinks). Each AS has the flexibility to decide on the aggregation granularity and hence the amount of state it wants to advertise. State is announced and exchanged in the form of a network state packet (nSP) similar to link state routing (see Fig. 4b). Each domain controller is responsible of creating the virtual topology of aNodes and vLinks and it is also responsible of propagating link information such as bandwidth, availability, variability and latency. The advantage of sharing information of aNodes and vLinks through nSPs is two fold. On the one hand, it allows each domain to customize the topology information to be shared with other domains. On the other hand, it provides useful information to other domains that can now decide how to route packets to get a given bandwidth, availability, variability and latency.

B. Dynamic creation of inter-domain tunnels using the GNRS

Next we describe a novel technique to setup cut-through switching tunnels across multiple domains that leverages the globally available name resolution service (GNRS). The main advantage of having a globally available entity is that the number of messages needed to be exchanged between domain controllers is significantly reduced.

To take full advantage of the GNRS, the routing framework names the tunnels. In other words, every tunnel created in a MobilityFirst network is an object that can be identified with a GUID (see Fig. 4a). When a domain controller initiates a request for a inter-domain tunnel, it contacts other domain controllers with a setup request. The request includes a label to identify traffic, as well as the GNRS entry containing information of the tunnel. When a neighbor domain controller accepts the request to create a tunnel, it creates a GNRS entry that contains information about the tunnel to be shared with other domain controllers. As a result, once the tunnel has been created, the domain controller that initiated the request knows the GUID of all entries needed to collect information about the tunnel.

Although some initial messages are needed to create the tunnel, one advantage of this technique is that tunnel maintenance and tear-down do not need further messaging. First, a domain controller can use the GNRS entry to share tunnel attributes with other domains, such as available bandwidth or


3. Inter-domain tunnel GNRS entry
GT<GT1,GT2,GT3>
GNRS
4. Intra-domain tunnels GNRS entries
GT1<AS1, attributes>
GT2<AS1, attributes>
GT3<AS1, attributes>
Domain controller
a2
aNode
vLink
2. Flow rule injection <in-port,label,out-port>
1. Path setup request/ACK
1. Path setup request/ACK
a1
a2
a3
a4
a5
a6
a7
a8
a9
a10
a11
Domain 1
Domain 2
Domain 3

Msg_Type	AS_Num:Source_aNode	Hop_to_Src
Internal Topologies:		
aNode#1-vLink<B,V,A,L>-aNode#2 aNode#2-vLink<B,V,A,L>-aNode#3 ... aNode#x-vLink<B,V,A,L>-aNode#y		
Neighbor Info:		
Neighbor_aNode#1-vLink<B,V,A,L> Neighbor_aNode#2-vLink<B,V,A,L> ... Neighbor_aNode#z-vLink<B,V,A,L>		

Fig. 4: **(a)** The steps needed to setup inter-domain tunnels. **(b)** The structure of network state packets.

expected time before having to terminate the tunnel. Second, terminating the tunnel is as simple as deleting the GNRS entry. Fig. 4a, suppose AS3 deletes the GNRS tunnel (GC) entry GT3. The initiating domain controller (AS1) notices that the entry has been deleted and concludes that AS3 is no longer part of the tunnel. Next, it deletes the GT entry known to all domains members of the tunnel. Finally, AS2 notices that the GT entry has been deleted. We evaluate the reduction of inter-domain messages in Section V-B.

C. *Dynamic traffic engineering based on flow behavior*

Given the knowledge of link conditions across domains and the ability to create inter-domain tunnels, the last responsibility of the domain controller is to identify flows to do traffic engineering. Indeed, the controller must find which flows will benefit the most from a inter-domain cut-through switching tunnel. We have developed traffic engineering techniques to detect elephant flows and support mobility, as well as user-requested cut-through switching, but we leave the explanation and evaluation of these for future work due to space limitations.

D. *Implementation techniques*

The current implementation of our framework is capable of cut-through switching at Layer 2 using virtual local area network (VLAN) tags, as described in Lara et al. [4]. However, the cut-through tunnels can also be implemented at lower layers, such as optical transport network (OTN) or wavelength division multiplexing (WDM). This allows for a much smarter bypassing technique, capable of deciding whether a bypass is more efficient at the optical layer or at the Ethernet layer. We will explore this in future work.

In the next section we focus on evaluating how inter-domain tunnels reduce the delay in the network and how the proposed framework simplifies how inter-domain tunnels are created.

V. Experimental evaluation

The evaluation section focuses on demonstrating how inter-domain tunnels reduce the number of packets that must be handled by domain controllers. We also compare the number of messages needed by the framework to create tunnels against a known protocol such as label distribution protocol (LDP) [8]. To do so, we provide results of the implementation on the GENI testbed [9] using the parameters shown in Table I and the topology shown in Fig. 5a.

TABLE I: Summary of components and key parameters used in the experiments

Type of switch	Open vSwitch version 1.9.3
Controller version	Floodlight 1.0
Controller host	Ubuntu 12.04 LTS
Controller host processor	Intel(R) Xeon(R), 2.67GHz
End-user OS	Ubuntu 12.04 LTS
Link bandwidth	100 Mbps

A. *Inter-domain tunneling and flow aggregation*

First, we show how multiple domains agreeing on an inter-domain label-based tunnel reduce the number of *packet_in* messages received by transit controllers. To do so, we randomly send 25 flows between AS1 and AS3.

Figure 5b shows the number of *packet_in* messages received per second by the controller. When all traffic is forwarded without using an inter-domain cut-through tunnel, the controller receives a total of 128 messages (top curve). However, when inter-domain tunnels are created for some of the flows, the total number of messages is reduced to 33 (bottom curve), for a 75% reduction. These results are specific to this topology and flow demands, but our goal is just to demonstrate how the creation of inter-domain tunnels can reduce the control plane delay.

B. *Reduction of label distribution messages*

Next we demonstrate how combining SDN with tunnel naming reduces the number of messages needed to create and maintain inter-domain tunnels. To do this, we briefly describe how all the functionality of LDP used in MPLS is implemented by our framework and we compare the number of messages needed to setup inter-domain tunnels.

Some benefits are due to using SDN. First, note that by using SDN, the number of intra-domain messages between peers is unnecessary. Instead, the SDN controller is responsible for pushing forwarding rules to the switches. Therefore, there is no need for intra-domain discovery messages. Second, session messages exist between domain controllers as opposed to peering routers. This reduces the number of messages needed because the only links carrying these messages are those between edge routers of neighbor domains. Third, advertisement messages are reduced for two reasons. On the one hand, intra-domain advertisement is not necessary because the domain controller is network aware. On the other hand, inter-domain advertisement is already achieved using the network state packets described in Section IV-A. For this reason, advertising messages are only needed to request a new tunnel

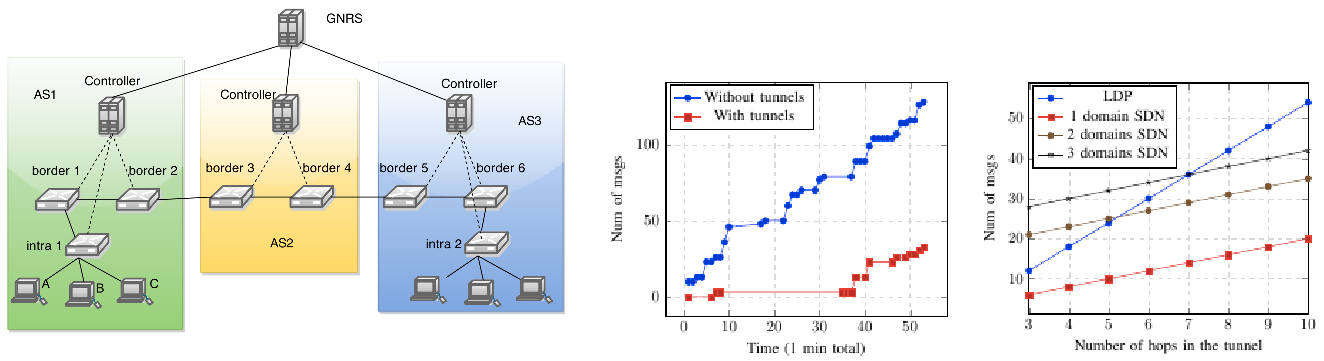


Fig. 5: **(a)** The experimental topology. Three SDN-based domains are deployed with end-nodes on ASes 1 and 3 and traffic going through an in-transit domain (AS2). **(b)** The accumulated number of *packet_in* messages received by the AS2 domain controller with and without inter-domain tunnels. **(c)** The number of messages needed to setup inter-domain tunnels using LDP or the proposed framework.

creation, as described in Section IV-B. Table II summarizes the key differences between messaging in LDP and the proposed framework.

TABLE II: Message equivalency between LDP and SDN-GNRS

Message type	LDP	SDN-GNRS (our work)
Discovery	Peer-to-peer between routers	No additional messages required, since this is achieved using network state packets
Session	Peer-to-peer between routers	A session between domain controllers is required regardless of tunnels. No additional messages required.
Advertisement	Peer-to-peer between routers	Controller-to-controller and controller-to-GNRS messages are required
Notification	Peer-to-peer between routers	Controller-to-controller messages are required
Flow rule injection	Not required	Controller-to-switch messages are needed to push forwarding rules to the switches. Figure 5c does consider these messages in the comparison.

Other benefits are due to naming the tunnels as network objects. First, the GNRS provides a common platform to exchange information between domain controllers. In MobilityFirst, the GNRS plays a key role in how packets are routed, so we can assume that it is a highly available entity and a session between each controller and the GNRS will exist. This reduces the complexity of establishing sessions between multiple domains. Second, relying on the GNRS reduces the number of inter-domain messages needed when more than two domains are involved. Suppose in Fig. 5a that AS3 needs to communicate with AS1. There is no need for AS2 to be involved in the communication and the GNRS provides a direct way for AS1 and AS3 to exchange information.

These benefits can be appreciated in Fig. 5c. First, notice how in LDP the number of messages grows independent of the number of domains traversed by the tunnel, as it requires three pairs of messages between peering routers in all cases. In contrast, when using SDN and the GNRS, the major factor for increase in the number of messages is the number of domains traversed. For a single domain, there is no need for the GNRS and the plot only includes messages needed to insert forwarding rules in the forwarding tables of the switches. Next, as the number of domains increases, we need more controller-to-controller messages as well as controller-to-GNRS messages. However, notice how the total number of messages stays below that of LDP for tunnels with four or more hops. A more thorough evaluation is needed to compare our solution with LDP. We only claim that, when counting the number of messages needed to establish a path, our solution requires fewer messages.

VI. Conclusion

This paper addressed the problem of dynamic creation of inter-domain cut-through switching tunnels in a named-based FIA such as MobilityFirst. We first motivated the need for inter-domain tunnels in SDN by modeling an optimization problem that minimizes the number of flows that must be handled by the domain controllers. To the best of our knowledge, this paper is the first to model the problem of inter-domain cut-through switching in SDN. Next, we presented a routing framework that enables inter-domain and intra-domain cut-through switching in the MobilityFirst architecture. The main novelties of this framework are the increased visibility of neighbor domains and the usage of a name resolution service to efficiently create and maintain inter-domain tunnels. The results show that in-transit domain controllers receive up to 75% less messages when using inter-domain tunnels. Likewise, our framework uses less messages than the traditional LDP protocol.

References

[1] D. Raychaudhuri *et al.*, "MobilityFirst: A Robust and Trustworthy Mobility-centric Architecture for the Future Internet," *SIGMOBILE Mob. Comput. Commun. Rev.*, vol. 16, no. 3, pp. 2–13, December 2012.
[2] L. Zhang *et al.*, "Named data networking," *SIGCOMM Comput. Commun. Rev.*, vol. 44, no. 3, pp. 66–73, 2014.
[3] T. Vu *et al.*, "DMap: A Shared Hosting Scheme for Dynamic Identifier to Locator Mappings in the Global Internet," in *IEEE Distributed Computing Systems (ICDCS)*, June 2012.
[4] A. Lara *et al.*, "Using OpenFlow to Provide Cut-Through Switching in MobilityFirst," *Photonic Network Communications*, vol. 28, no. 2, pp. 165–177, 2014.
[5] A. Tomaszewski *et al.*, "Distributed inter-domain link capacity optimization for inter-domain IP/MPLS routing," in *Proceedings of IEEE GLOBECOM*. IEEE, 2007.
[6] M. Roughan *et al.*, "GATEway: symbiotic inter-domain traffic engineering," *Telecommunication Systems*, vol. 47, no. 1-2, pp. 3–17, 2011.
[7] M. Chamania *et al.*, "Effective usage of dynamic circuits for IP routing," in *Proceedings of IEEE ICC*, 2010.
[8] "Label Distribution Protocol RFC," https://tools.ietf.org/html/rfc5036.
[9] M. Berman *et al.*, "GENI: a federated testbed for innovative network experiments," *Computer Networks*, vol. 61, pp. 5–23, 2014.