Formal Approach for Resilient Reachability based on End-System Route Agility

Usman Rauf
urauf@uncc.edu
UNC Charlotte
Charlotte, NC. USA

Fida Gillani
sgillan4@uncc.edu
UNC Charlotte
Charlotte, NC. USA

Ehab Al-Shaer
ealshaer@uncc.edu
UNC Charlotte
Charlotte, NC. USA

Mahantesh Halappanavar
PNNL, Richland, WA. USA
mahantesh.halappanavar@pnnl.org

Samrat Chatterjee
PNNL, Richland, WA. USA
samrat.chatterjee@pnnl.org

Christopher Oehmen
PNNL, Richland, WA. USA
christopher.oehmen@pnnl.org

ABSTRACT

The deterministic nature of existing routing protocols has resulted into an ossified Internet with static and predictable network routes. This gives persistent attackers (e.g. eavesdroppers and DDoS attackers) plenty of time to study the network and identify the vulnerable (critical) links to plan devastating and stealthy attacks. Recently, Moving Target Defense (MTD) based approaches have been proposed to to defend against DoS attacks. However, MTD based approaches for route mutation are oriented towards reconfiguring the parameters in Local Area Networks (LANs), and do not provide any protection against infrastructure level attacks, which inherently limits their use for mission critical services over the Internet infrastructure. To cope with these issues, we extend the current routing architecture to consider end-hosts as routing elements, and present a formal method based agile defense mechanism to embed resiliency in the existing cyber infrastructure. The major contributions of this paper include: (1) formalization of efficient and resilient End to End (E2E) reachability problem as a constraint satisfaction problem, which identifies the potential end-hosts to reach a destination while satisfying resilience and QoS constraints, (2) design and implementation of a novel decentralized End Point Route Mutation (EPRM) protocol, and (3) design and implementation of planning algorithm to minimize the overlap between multiple flows, for the sake of maximizing the agility in the system. Our PlanetLab based implementation and evaluation validates the correctness, effectiveness and scalability of the proposed approach.

1. INTRODUCTION

The Internet has transformed the world into a global village where even small to moderate enterprises have offices scattered throughout the world. These enterprises often communicate critical information of high business value with each other. The confidentiality, integrity and availability (CIA) of this information are essential for the survival of such businesses.

Over the years, with the advancements in encryption and hashing standards, even a small enterprise can manage and ensure the end-to-end confidentiality and integrity of their critical information. However, they do not have the same level of freedom and control to ensure the availability of this critical information across the Internet. As this information traverses geographical boundaries spanning over multiple autonomous systems (ASes), coordinating a controlled and manageable routing is practically not possible. Therefore, in case of any device or link failure along the network path, due to faults or cyber attacks, leaves such enterprises helpless and they have to resort on the slow Internet recovery process to restore the availability of their critical information. This paper aims to empower enterprises to ensure the end-to-end availability of their critical communications under highly susceptible situations.

Recently, incidents challenging the network availability are on the rise especially, the scale and sophistication of Distributed Denial of Service (DDoS) attacks have been rapidly increasing. More of this is attributed to the Internet ossification that has resulted into some inherent vulnerabilities. First, network paths are mostly static which makes network reconnaissance for identifying links-to-destination association feasible with low-cost [23]. Second, the power-law distribution of traffic flows over network links results into emergence of critical links, i.e., to any destination only a few links carry majority of its traffic [7]. Third, a destination shares its critical links with its geographical neighbors [23].

These fundamental design level vulnerabilities allow an adversary to easily learn and compute a set of critical links and plan devastating attacks to isolate the traffic between entire regions/states [23][27]. This imposes severe threats to critical infrastructures, and demands novel approaches for resilient cyber agility to defend against infrastructure level attacks.

Contributions: The main contribution of this paper is to develop a resilient routing function as a service that offers an efficient and decentralize route mutation without requiring to reconfigure the infrastructure devices (e.g., routers

MTD'16, October 24 2016, Vienna, Austria

DOI: http://dx.doi.org/10.1145/2995272.2995275

which belong to ISPs, and end-user have no control over them). Our resilient routing function employs end-hosts based virtual routers to increase the routing diversity and mutation space while reducing computational overhead. Our approach does not require infrastructure level support (i.e., configuring routers) and is not centralized, hence, does not pose any significant overhead.

Our approach assumes that there exists end-hosts based virtual routers (called peers) in the network. A practical example of such an architecture is Planetlab [5] that we leverage in our proposed approach. PlanetLab is a publicly available prominent virtualized infrastructure in which the physical network resources are geographically distributed across hundreds of physical domains (mostly universities) on the Internet. Any user from a participating institute can request a slice on PlanetLab. She can select a set of PlanetLab nodes (peers) around the world and add these to her slice. She can connect these peers together through tunnels to form a virtual network (VN) within the slice. Alternatively, such an infrastructure can easily be created by strategically placing peers in clouds at different places without worrying about high operational and capital cost, and there exists some industrial solutions leveraging such an architecture but for different purposes [6]. In our approach, a source leverages this architecture to establish different resilient network paths (virtual paths) for its critical flows while satisfying QoS and resilience requirements, as illustrated in Fig. 1.

We formalize this end-to-end (E2E) resilient reachability problem as a constraint satisfaction problem [8, 9], using Satisfiability Modulo Theories (SMT) [12]. This model identifies the potential virtual paths to ensure reachability between a source-destination pair with significantly low time overhead as compared to the earlier formalizations. We present a planning algorithm to minimize the overlap/ intersection of physical links and peers between multiple flows, to maximize the agility in the system. We have used PlanetLab to implementation and evaluate the correctness and effectiveness of our approach. Moreover, we have used peer to peer data sets available at SNAP [11] to evaluate the scalability of our approach in large networks.

Rest of the organization for the article is as follows. Section 2 presents the related work and limitations of existing approaches. Section 3, describes the main problem, its hardness and adversary model followed by technical details in Section 4. Section 5 evaluates effectiveness of the proposed approach using Planetlab and simulation based experiments. Section 6 summarizes the contributions of the paper and discusses some potential extensions of our proposed approach.

2. RELATED WORK

The current architecture of routing protocols is static and deterministic, which yields predictable configurations for end-to-end reachability and provides significant advantages to adversaries. Such vulnerabilities can lead to devastating infrastructure level attacks [23]. Over the recent years, the research community has developed interesting solutions to eliminate such vulnerabilities. However, these existing approaches exhibit certain limitations that restrain their use for mission critical services.

One proposed approach resides in the realm of Dynamic Routing (DR). The purpose of DR is to offer reliable transmission and load balancing for mission critical services. However, existing practices are predictable and vulnerable against eavesdropping and infrastructure level DDoS attacks [24]. We show that the threat-aware randomness introduced by our approach avoids these limitations.

The concept of randomization for multi-path routing in wireless networks, to avoid jamming or black hole attacks, is also not old, however, the approach is far from being practical for wired networks [19]. As it is almost infeasible to find solution for a random walk based algorithm which satisfies multiple Quality of Service (QoS) and resilience constraints simultaneously [3].

Similarly, the first effort in the domain of proactive Moving Target Defense (MTD) which involves the idea of random multi-path routing (named as: Random Route Mutation (RRM)) for wired networks, was proposed by Qi et al. [3]. The authors proposed that proactively changing the route between source and destination by re-configuring intermediate routers, can reduce the potential damage caused by infrastructure level attacks. The authors also integrated game-theoretic approach to argue about the effectiveness of RRM against persistent attackers [4]. Another similar approach has been presented recently based on agile virtual networks, which claims to make networks more resilient by dynamically changing the virtual network mapping of the underlying substrate network [2].

The main limitations of these randomization and mutation based techniques are that: (1) they require infrastructure level support, i.e, configuring and updating routers repetitively after certain period of time, (2) they are centralized, hence prone to single point of failure, (3) the overhead for the proposed formalizations is significantly high, as the whole computation is done by the central controller, which increases the computational complexity with the increase in network size, and (4) these techniques assume the availability of many disjoint routes in the network, which may not be the case in real networks.

Other overlay based reactive MTD architectures [20, 21] leverage cloud environments to host a small set of active proxies, to hide the end-host/application server during end-to-end communication. Incoming connection requests are validated by a well provisioned lookup server which then relays each authorized user to one of the secret active proxies to serve the user‘s subsequent requests. When under attack, a central server instantiates new proxies and clients associated with attacked proxies are migrated to these newly instantiated proxies to avoid/reduce the impact of the attack. The major limitations of these approaches are, (1) they only provides anonymity to the application while hiding it behind the pool of proxies, regardless of available bandwidth assigns proxies to the users randomly, (2) centralized and computationally very expensive, and (3) once attacked, it simply initiates new proxy servers rather than avoiding the critical links which may be under attacked and do not provide any defense against infrastructure level attacks.

Another approach which is erroneously considered as an alternative to MTD approaches is The onion Routing (ToR), developed by the ToR Project [28]. However, ToR only provides anonymity and privacy for end-to-end communication over public internet without any regard to available bandwidth. It is static as once random set of peers are selected, during a session the assigned peers do not change, which makes it vulnerable to sniffing and highly sophisticated DDoS attacks like, Crossfire [23] and Coremelt [27]. It does not take care of the geographical locations of the

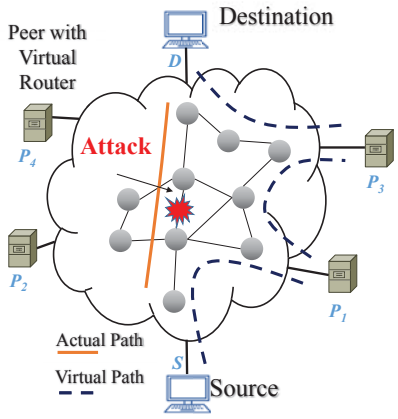

Figure 1: Concept of End Point Route Agility

peers while peer selection process, as path between source and destination is not more than a random walk. Realization of geographical locations of peers, while peer selection process, is highly important for reactive defense techniques (to avoid the critical links).

3. PROBLEM DEFINITION AND ADVERSARY MODEL

Traditional networks are modeled as a graph *G(V, E)*, where V is a set for vertices (in this case end-hosts) and E is a set of edges (in this case intermediate peers and links among them). Therefore, corresponding to traditional networks a placement of EPRM problem can be considered as a five tuple T=(S, D, P, C_i, Lab), where;

- $S \in V$ is source for a certain flow
- $D \in V$ is destination for a certain flow
- $P \subset V$ is a set of potential peers, such that: $S \cap P : \emptyset$ and $D \cap P : \emptyset$ to avoid the cycles
- C_i is a set of connectivity constraints for a pair (S, D)
- Lab is a labeling function such that $Lab(S, D, C_i, P) : (S, D) \longrightarrow P_i$ where $P_i \subseteq P$

Figure 1 provides an overview of our proposed approach for routing agility. A virtual path is comprised of suitable peers (P_i) with a specific order and changing the order results into a different virtual path. The goal of EPRM is to efficiently compute the set virtual paths for critical communications and then decide a sequence to use them for a source-destination pair, while satisfying following constraints.

- ***QoS Constraints***: The selected mutation set (MS) is the set of all valid virtual paths for a source-destination pair. The MS should meet QoS requirements such as; available bandwidth of all virtual paths must be greater than the traffic load generated by the source, number of hops should be limited and the E2E delay must be comparable to the delay of the actual physical path of the source-destination pair (Fig. 1).
- ***Resilience Constraint***: To increase unpredictability or chaos for the attacker, all virtual paths in a sequence must have minimum (link) overlap with the immediate neighboring virtual paths.

3.1 Hardness of Problem

E2E resilient reachability of a single flow/transmission requires the upper bound on the number of peers to be used to reach the destination, which according to the theory [14] is a NP-complete problem and cannot be solved in polynomial time, but the efficiency is highly dependent on the value of *Upper Bound (UB)*. In current architecture, we use *UB=3* as an upper bound, which means a transmission/flow can pass through at-most three intermediate peers or in other words, a virtual path cannot have length greater than 3 hops. Our evaluation shows that only 3 hops can add enough variance to evade and deter persistent attackers.

3.2 Attack Model

For DDoS attacks, we assume that an attacker can disrupt/compromise limited number of links$\subset$*critical_set*, which she believes to be critical, for a specific period of time. For selection of target our attack model follows uniform probability distribution to select the the target link. There can be two types of attackers. *Naive Attacker*: An attacker who does not have any information about the critical links and nodes, and attacks the network in a random fashion for a random amount of time without any prior information about the flows. *Sophisticated Attacker*: An attacker who can gain information about the critical links and nodes in the network, and can launch the attack accordingly to maximize her devastation. She can also attack randomly on different critical links and move the attack surface over these critical network links to avoid detection and to harden the recovery.

4. TECHNICAL APPROACH

4.1 Challenges

The main challenge in EPRM protocol is to determine a correct-by-construction mutation set (set of virtual paths), given a pair of source-destination, and then efficiently mutate the virtual path, on the fly during multiple transmissions, to disable the capabilities of an attacker for launching DDoS attack on specific links or nodes. We do not rely on configuring infrastructure level devices for monitoring or forwarding packets through a desired path.

Mostly, all end-hosts use TCP based communication channel and the physical path between them relatively stays stable for almost 80% of the time [23]. In TCP sessions, both source and destination use TCP congestion window to decide how much data should be transmitted and using this transmission history between any two peers, we can easily estimate the safe available bandwidth of the Internet path connecting these peers, corresponding to different periods of time. This is achieved by deploying an agent (virtual router) on every peer, which shares this information to a source upon a request.

As a result, the source can alter the sequence or narrow down its choices of virtual paths to be used based on QoS constraints. To reduce the overhead, mutation sets are precomputed according to reachability constraints and their sequences are evaluated on run time. We use SMT to formalize the problem and the constraints as uninterpreted coupled functions in first order logic. Evaluation of these function as true means that there exist an interpretation for the first order logical formula which satisfies the QoS and resilience constraints. In order to allow smooth transition between virtual paths (mutations), configuration changes are carefully

observed and managed to avoid disruption between multiple transmissions.

4.2 SMT Formalization of EPRM

4.2.1 Formalization of E2E Resilient Reachability

The formalization of E2E resilient reachability is fundamental to EPRM protocol for calculating the mutation set for multiple flows. The problem of computing set of routes with all possible length, is computationally expensive. Therefore, to make this approach practical for usage we put an upper bound on the number of intermediate peers to be used, for calculation of reachability, such that the problem can be solve in polynomial time. The most important ingredient of virtual reachability is logical path connectivity, which we define as a boolean function that accepts two arguments and maps it to true/false if two nodes are logically connected/not connected.

$$L(x, y) : \mathbb{N}^2 \mapsto \mathbb{B}$$

Where x and y are source and destination in set of peers. Using the same notion we can define virtual reachability between two nodes as a conjunction of multiple constraints:

$$R(x, y) : \mathbb{N}^2 \mapsto \mathbb{B}$$
$$R(x, y) \rightarrow \left(\bigwedge_{z=1}^{4} \Psi_z \right)$$

The above relation states that; there exists a virtual path between x and y if following constraints (Ψ_i) hold true.

$$\Psi_1 = \forall_{x,y} \Big(\exists_{i,j,k} \Big(L(x,i) \wedge L(i,j) \wedge L(j,k) \wedge L(k,y) \Big) \vee \exists_{i,j} \Big(L(x,i) \wedge L(i,j) \wedge L(j,y) \Big) \vee \exists_{i} \Big(L(x,i) \wedge L(i,y) \Big) \Big)$$

$$\Psi_2 = \forall_x \Big(\neg L(x,x) \Big)$$

$$\Psi_3 = \forall_{x,y} \Big(L(x,y) \rightarrow L(y,x) \Big)$$

$$\Psi_4 = \forall_u \Big(\neg L(u,x) \Big) \wedge \forall_v \Big(\neg L(y,v) \Big) \; s.t. \; x = source \; and \; y = destination$$

Note that virtual path is bidirectional: $R(x, y) \leftrightarrow R(y, x)$. Ψ_1 is the most fundamental property, which defines the characteristic of bounded reachability. We put an upper bound on the length through Ψ_1, by restricting the maximum number of intermediate peers to be '3'. Therefore, using our reachability formalization a peer *'x'* (source) can use at most '3' intermediate peers (*i,j,k*) to reach the destination *'y'*, under critical/normal circumstances.

The value of upper bound is variable and can be changed as per user requirements. This does not effect the protocol structure, and only increases the length of formalization. Where *i,j* and *k* can assume any value in peers set P_i, as far as connectivity constraints/requirements are satisfied. Ψ_2 implies that no node should be reachable to itself directly, to avoid infinite length self-cycles. Ψ_3 implies that logical path and reachability is a bidirectional property. If *'y'* is reachable by *'x'*, then *'x'* is also reachable by *'y'* in terms of connectivity. Finally, Ψ_4 implies that while considering *'x'* as a source and *'y'* as destination, avoid all irrelevant and useless interpretations (to reduce computational complexity) of the function *R(x,y)* in which *'x'* is not source and *'y'* is not destination. In simpler words this property is responsible for fixing the source and destination to reduce time and space complexity. Note that this is a generalized definition for computing reachable set from multiple source to multiple destinations as *'x'* and *'y'* are variable.

4.2.2 Formalization of QoS Constraints

As we discussed earlier, QoS constraint is strongly subjected to bounded delays (nodal processing time), maximum number of hops to be used for a transmission and link available bandwidth for quality transmission. This inherently restricts reachability formalization to only find a path which may only have at most certain number of hops (Θ_{hops}), certain available band-width (which can accommodate desired transfer rate: Θ_{trans_rate}) and certain amount of load (Θ_{load}) on intermediate peers. Where Θ_{hops}, Θ_{trans_rate} and Θ_{load} are variable threshold values, provided by the user, for desired maximum number of hops, desired transfer rate (which is directly related to the quality of transmission), and nodal processing time/delay respectively. These threshold values are service specific, and depends on the quality demanded by the service running on a specific host.

Formalization of Intermediate Hops Constraint: The formal definition of QoS constraint, for maximum number of intermediate hops to be used, is as follows:

$$QoS_{inter\_hops} : \forall_{x,y} \Big(\exists_{i,j,k} \Big(\#_{hops} \leq \Theta_{hops} \Big) \Big)$$

where $\#_{hops}$ can be defined as follows:

$$\#_{hops} : \Big(\sum \Big(H(x,i), H(i,j), H(j,k), H(k,y) \Big) \vee \sum \Big(H(x,i), H(i,j), H(j,y) \Big) \vee \sum \Big(H(x,i), H(i,y) \Big) \Big)$$

The uninterpreted function $H : \mathbb{N}^2 \mapsto \mathbb{N}$ represents the number of hops between a pair of source and destination, therefore, it accepts two arguments. $\#_{hops}$ represents the maximum number of hops, regardless of how many intermediate peers have been chosen to reach a certain destination. The above mention constraint is a generalized version, as the proposed approach can decide to choose one, two, or three intermediate peers, depending on the situation and availability, which results in the form of a disjunction of three different cases. The proposed formalization limits that the number of hops on chosen route, in any case, should be at most Θ_{hops} for any pair of source and destination.

Formalization of Link band-width Constraint: Following is the formal representation of link band-width constraint:

$$l^a(x,y) : \mathbb{N}^2 \mapsto \mathbb{R}$$

$$QoS_{band\_width} : \forall_{x,y}\left(\bigvee_{i:1}^{3}\left(\Phi_i\right)\right)$$

$$\Phi_1 : \exists_{i,j,k}\Big(l^a(x,i) > \Theta_{trans\_rate} \wedge l^a(i,j) > \Theta_{trans\_rate} \wedge l^a(j,k) > \Theta_{trans\_rate} \wedge l^a(k,y) > \Theta_{trans\_rate}\Big)$$

$$\Phi_2 : \exists_{i,j}\Big(l^a(x,i) > \Theta_{trans\_rate} \wedge l^a(i,j) > \Theta_{trans\_rate} \wedge l^a(j,y) > \Theta_{trans\_rate}\Big)$$

$$\Phi_3 : \exists_{i}\Big(l^a(x,i) > \Theta_{trans\_rate} \wedge l^a(i,y) > \Theta_{trans\_rate}\Big)$$

To avoid a overloaded links, which may be under attack, link available bandwidth constraint is very effective. This constraint must be checked on run time (online) before selecting a sequence of peers. The main reason for checking this constraint online/on run-time is that the peers may be off-line at any instance of time and thus making them unable to process a request, therefore, this is one of few main steps in protocol which must be checked online. Formal description of online-phase of the protocol is explained in the next section. The above mentioned constraint is a generalized version, which covers a path of length three. For better understanding of the readers it can easily be broken into three parts by removing the disjunction symbol. $l^a(x,y)$ refers to the cumulative link available bandwidth between peer x and y (including all hops) and Θ_{trans_rate} refers to the desired transfer rate of the source. Therefore, at any given time the algorithm must find a satisfiable path/route for which the available bandwidth is more than the desired transfer rate of the source.

Formalization of Load Constraint: Formal definition of load constraint on intermediate peers is as follows:

$$l(u) : \mathbb{N} \mapsto \mathbb{R}$$

$$Total_{load} : \sum\left(l(i), l(j), l(k)\right)$$

$$QoS_{load} : \forall_{x,y}\left(\exists_{i,j,k}\left(Total_{load} \leq \Theta_{load}\right)\right)$$

Where *l(u)* is a function which evaluates load on an intermediate peer. This *load* constraint implies that only those intermediate peers should be selected which have collective load less then certain threshold Θ_{load} to maintain quality of transmission and control delays.

4.3 Formal Description of EPRM Protocol

In this section we formally describe the structure of the protocol along with the algorithm of proposed approach and some basic definitions which are fundamental for better understanding of the readers.

Definition 1. ***Potential Peers Set (PPS):*** *A PPS is a set of possible logical paths for a pair of source and destination satisfying reachability criteria. PPS for a network in Fig.2 contains:* $\{(P_1,P_2,P_3),(P_4,P_5,P_6),(P_7,P_8,P_9),...\}$

Definition 2. ***Mutation Set (MS):*** *A MS is a set of logical paths (constituting peers) which meet the aforementioned constraints in order to connect a pair of source and destination, such that MS⊆ PPS. An instance of MS is a logical path* $L_i \in MS$. *MS from an example network given in Fig. 2 is,* $\{L_1 : (P_1,P_2,P_3),\ L_2 : (P_4,P_5,P_6),\ L_3 = (P_7,P_8,P_9)\}$

Definition 3. ***Mutation Sequence (MSeq):*** *A MSeq is a set with an ordered sequence of mutations (logical paths) for a pair of source and destination, such that:* $MSeq \subseteq MS$, *which is calculated at run-time based on the real-time available bandwidth updates from the peers.*

Definition 4. ***Mutation Cycle:*** *A time duration after which a logical path repeats, once every possible member of MSeq is used, is referred as Mutation Cycle duration.*

4.3.1 Protocol Description

Off-line Phase: Formation of Mutation Set:

The first and foremost step of EPRM protocol is to efficiently compute the mutation set given a set of source and destination according QoS requirements. For our implementation and evaluation we consider E2E reachability using at most three intermediate peers (length=3) and present the corresponding Algo. 1 to compute mutation set[1], considering three intermediate peers (length=3). Nevertheless, the algorithm can easily be extended and generalized up to any desired length.

Online Phase: Calculation of Mutation Sequence:

- *Broadcast Association Information:* Once a source computes MS, it broadcasts to all of potential peers in MS, an activation order/sequence.
- *Liveness Response:* Immediately after receiving activation order/sequence each selected peer must send available path bandwidth and information/packet drop rate, from itself to its successor and next peer to the source. This information is used to finalize the mutation sequence (MSeq).
- *Calculating Mutation Sequence:* Based on the information received in liveness response phase, the mutation sequence of active intermediate peers is efficiently calculated for every flow (or a set of flows) such that: $L_i \in$ MSeq. Moreover, every logical path in MSeq exhibit minimum overlapping with its predecessor and successor logical paths. Furthermore, if there exist a set of some critical physical links (β) that must be avoided, our approach avoids all such logical paths that uses these critical links. This phase can be repeated iteratively, until the mutation sequences start repeating or according to user requirement for a certain number of flows. Algorithm 2 also guarantees that the flow distribution of each peer and intermediate link will be same if there are only disjoint routes in the network.

***Discussion*:** Our approach involves two phase mutations which makes it difficult and chaotic for an attacker to learn about insights of the network. The first phase mutation is the selection of best suited candidates for a MS, which can be shuffled and re-selected after a certain time in an

[1]We use iterative algorithm in actual and this recursion based algorithm (Algo. 2) is presented here for simplicity and limitation of space.

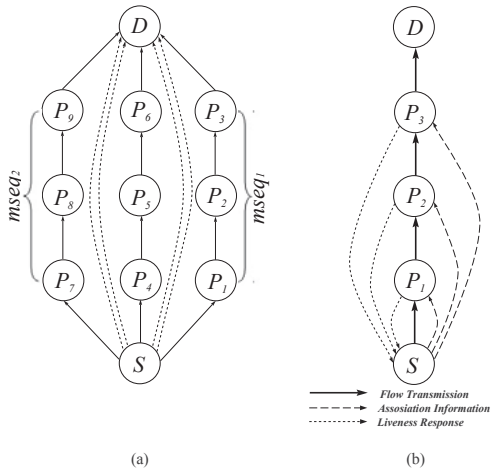

Figure 2: (a) EPRM flow transmission using multiple MSeq of intermediate peers. (b) Schematic diagram for interaction between source and elements of MS.

Algorithm 1 Computing Mutation Set

```
1: P = {(P_i), (P_i, P_j), (P_i, P_j, P_k)}
2: C = {  Set of constraints  }
3: procedure LAB(P,C)
4:     (result, model) = check_satisfiability(P, C)
5:     if result==SAT then
6:         MS.insert(model)
7:         P' = model.P
8:         for each peer p ∈ P' do
9:             assertion_stack.push()
10:            C.insert(¬p)
11:            Lab(P, C)
12:        end for
13:    else
14:        assertion_stack.pop()
15:        return
16:    end if
17: end procedure
```

iterative way. The second phase mutation becomes active when different mutation sequences ($mseq_i$) are selected from MS for different set of flows in an iterative manner. Fig. 2 presents a schematic diagram of interaction between source and other selected peers in order to obtain information from neighboring peers.

One of the main goals in MTD research, is to minimize the overlap of physical paths between consecutive flows. Reducing the overlap will increase the agility in the system, as the traffic will be evenly distributed in the network, making it harder for an attacker to identify critical links. We also present an algorithm in Algo. 2 to minimizing path overlapping between multiple flows. Unlike previous methods [3, 4], we do not completely negate the previously used logical paths. Instead, we randomize the mutation cycle of a logical path to uniformly distribute its reuse probability. Every time a logical path is inserted into MSeq (Line-13, Algo. 2), it is assigned a random counter t_i. For each insertion in MSeq, this counter decrements for all logical paths already inserted (Line-16). When the counter t_i expires (becomes zero) for a logical path L_i then this logical path becomes a contender to be inserted back in MSeq provided all other conditions are satisfied. To the best of our knowledge this is the first run time optimization algorithm for minimizing the overlap between multiple flows in a network using SMT in the domain of MTD research.

In Algo. 2, we use *Route* to store list of physical links along a logical path and *findLinks()* function uses `traceroute` to find these links. The *physical_path* variable represents a set of physical links along the physical path between source and destination. The *If* condition (Line 20) ensures that if there exists any logical path that was skipped in the last iteration due to the overlapping condition, it can be checked again to see if it becomes valid again in the new sequence.

Algorithm 2 Computing Mutation Sequence

```
1: procedure FINDNEXT
2:     Route = ∅
3:     for each logical_path L_i ∈ MS do
4:         Route.insert(findLinks(L_i))    ▷ Find links along the logical path.
5:     end for
6:     prev = physical_path
7:     MSeq = ∅
8:     iteration = 0
9:     while True do
10:        for each route r_i ∈ Route do
11:            ratio = |r_i ∩ prev| / |r_i|    ▷ |r_i| counts set elements.
12:            if (r_i ∉ MSeq or t_i = 0) and (ratio < Θ_ratio) and (β ∩ r_i = ∅) and (Total^i_load ≤ Θ^i_load) then
13:                MSeq.insert(L_i)
14:                prev = r_i
15:                t_i = Rand(Θ_length)
16:                Decrement all other t
17:            end if
18:        end for
19:        iteration = iteration + 1
20:        if count(Selected) ≥ Θ_length then
21:            break While Loop
22:        end if
23:        if iteration ≥ Θ_reduce_ratio then
24:            Reduce Θ_ratio  ▷ Allowing more overlapping.
25:        end if
26:    end while
27: end procedure
```

4.4 Complexity Analysis of EPRM

For a fixed source and destination if we only want to find one satisfiable path, then the complexity (either the worst case or the average case) of Dijkstra algorithm is $O(n^2)$, and the average complexity algorithm in the paper is $O(\Delta^\eta)$, where Δ is the maximum node degree of the graph (physical topology), and η is the upper bound of the length (in terms of hops) between two peers in the graph. For most practical topologies, Δ and η are small numbers. If one wants to enumerate all satisfiable paths, then the possible number of paths is $O(n!)$ but the our approach in the paper has complexity $O(n^3)$, since one needs to test all possible combinations of three intermediate peers and it surprisingly performs batter than Dijkstra.

5. EVALUATION AND EFFECTIVENESS OF EPRM

5.1 PlanetLab Setup

In our design and evaluations of EPRM, we have selected to implement EPRM architecture in Planetlab. As mentioned earlier, each user is given a slice in Planetlab and user can add nodes in this slice from all over the world. We use this node as a peer that hosts a virtual router. We create a virtual network that connects a source and its destinations by creating tunnels from source to its peers and between peers (Fig. 1).

There are two common methods to create and manage a VN within a slice on PlanetLab. The first implements VNs in user space. This requires the creation of a virtual router in each peer using the *Click* router and connecting the virtual routers together with UDP tunnels. User space implementations increase the latency of forwarding packets due to copying the packets between kernel and user space, and also the time to wait for Click to run on the CPU. Therefore, we implement VNs in kernel space by setting up packet forwarding in kernel space using the Vsys API [26]. With Vsys, a virtual router in a PlanetLab node can install routes to the forwarding table in the kernel space. When a PlanetLab node receives packets for one of the slices it hosts, it does not copy the packets from kernel space to user space before the redirection. Instead, the packets are forwarded directly according to the forwarding table of the node. Thus, the latency of packet forwarding within a node is considerably lower compare to using the user space routers.

Vsys currently is fully functional on the PlanetLab Europe nodes. It allows each slice to have access to certain super user sudo privileges, that includes creating virtual interfaces, tunnels and forwarding table entries for that slice. Each user is assigned a private IP subnet for the slice and she can only modify the forwarding table entries that belong to her assigned IP subnet. With this kernel space forwarding table sharing feature, a user space router can install routes to the forwarding table in kernel space directly. When a PlanetLab node receives a packet from one of its virtual interfaces, the node checks for the best route of the packetŠs source IP address in the forwarding table to forward the packet accordingly.

To setup VNs, we use a Python Vsys API package provided by NEPI [25]. According to the example topology, given in Fig. 1, we connect the PlanetLab nodes with point-to-point tunnels through the Vsys API. We assign private IP addresses within the assigned subnet of our slice to the virtual interfaces for those tunnels. Then we install the precomputed routing table entries (from Algo. 1) to the forwarding tables of the PlanetLab nodes through the Vsys API and use Algo. 2 to change their priorities to achieve desired sequence.

5.2 Simulation Setup

For SMT formalization, we use Z3 solver [10], and for evaluation we use real life peer to peer data sets available at SNAP [11]. As the real life data sets are not sufficient in numbers, we also use BRITE [16] as network generator to generate additional topologies closer to real life characteristics. In BRITE we use the Waxman model to generate random and power law based preferential attachments. The two parameters used for Waxman model are $\alpha = 0.2$ and $\beta = 0.15$ and the network growth type is set to be incremental. All the experiments are conducted on Core i7 machines with 3.4GHz processor, and 16GB memory.

5.3 Effectiveness and Resilience of EPRM

The ideal metric available in literature to analyze the effectiveness of the MTD or route mutation approaches is (*Mutation Protection Effectiveness (MPE)*)[3] (which assumes that defender is following Shamir's criteria [18]). *MPE* metric is represented as follows:

$$MPE = 1 - \sum_{1=1}^{m} \frac{R}{N} \frac{1}{M} d_i$$

MPE can be defined to be the percentage of packets in a transmission which do not pass through any intermediate nodes (links) that are being compromised or eavesdropped. Note that this metric is application specific and minimum requirement of MPE can vary from application to application. MPE calculates the average case effectiveness of a deployed mechanism, which is why it is appropriate to compare between different route mutation techniques. This metric assumes that defender and attacker are randomly choosing routes regardless of the actions taken by each other.

In our Planetlab based experiment, we calculate the physical topology using `traceroute` probes and used this in our experiments. Attacker can target R routes out of N possible disjoint routes, in M mutation intervals; where d_i is the damage caused by the degradation of the flows within the same interval. Suppose if attacker is targeting same routes which are being used by the defender in a certain interval, and this causes a degradation of *40%* traffic/flows, then d_i can be 0.4.

Our approach is rather mixture of proactive and reactive mechanisms as we mutate peers in a proactive fashion, but after collecting information from our potential peers; and based on that we take decision about which links to avoid (if under attack) to maintain QoS, this highlights the resilient features of the proposed strategy. We setup our experiment such that, an adversary has limited knowledge about critical links and limited resources (an adversary has capacity to cause maximum *80%* degradation in traffic/flows). We cannot assume an adversary model with infinite resources as this is not a practical assumption. We target/attack different percentages of routes/intermidiate links (R) in *Mutation Set* (as adversary has knowledge about critical links) to observe the effectiveness of EPRM. We also assume that the adversary's behavior is dynamic, but in a certain interval he/she is only targeting certain percentage of links.

We compare the effectiveness of our proposed approach against RRM and non-RRM techniques. We choose network with 100 nodes and 256 edges (with length L=3), and run experiments for single flow and multiple flows. Fig. 3 (a) shows the MPE effectiveness of EPRM, in comparison with RRM and non-RRM approaches (for the network of same characteristics). For small number of flows EPRM seems to have significant benefit over RRM, and non-RRM approaches. It is 20% more efficient than RRM (when L=3 and flows=5) and 51% more efficient than tradition non-RRM approaches (with L=3 and flows=5, when 25% of the network is under attack). The effectiveness of EPRM seems to decrease as number of flows with-in the network increase. This justifies the fact that potentially EPRM can be considered as an ideal candidate for mission critical services. Al-

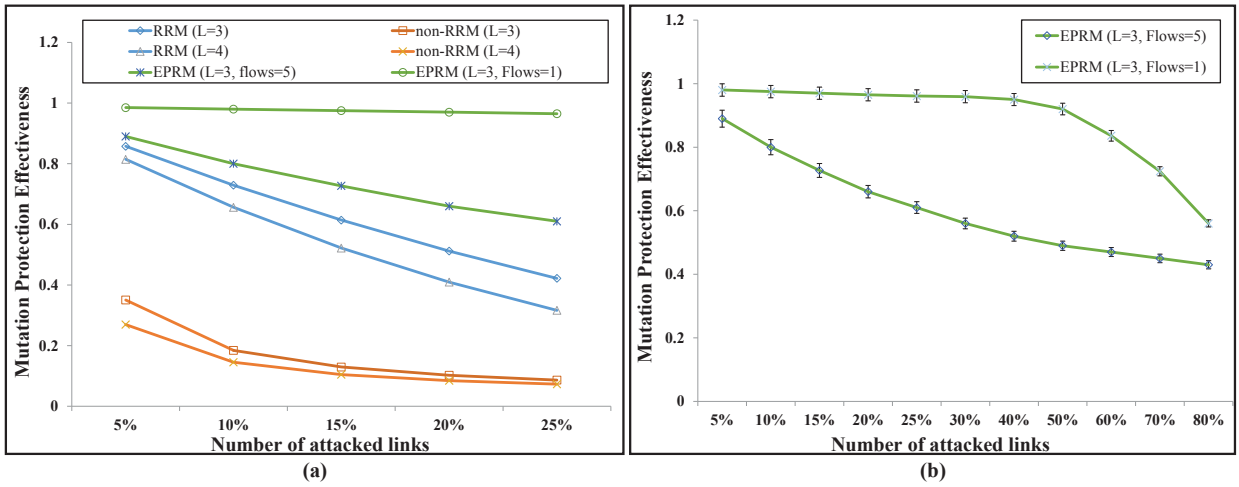

Figure 3: (a) MPE comparison of RRM, non-RRM, and EPRM; (b) Extended MPE of EPRM for Single and multiple-flows

though it can operate in proactive mode, but as the network size becomes larger (which is the case in real life), its overhead can increase. Therefore, the ideal operational choice for EPRM is its utilization for mission critical services in defensive/reactive mode.

Figure. 3 (b) shows the extended evaluation of EPRM, where even if *80%* links are under attack, more than *45%* flows are safely transmitted for multiple flow mode as far as network is not fragmented, which potentially means unless adversary attacks the whole infrastructure. The proposed approach reactively changes a path/route if there is a blockage in any intermediate link. The *5%* degradation of flows in single-mode experiment is due to the reaction time of the approach, which is solely dependent on how frequently the members of *Selected Mutation Set* send update to the source (about the drop rate from them to their successors). Assuming that adversary is also mutating attacked links ($attack\_set \subset MS$ at any interval) will not reduce the effectiveness, as online phase of the proposed approach have tendency to rapidly mutate between multiple choices, unless $attack\_set \supseteq MS$, which means the whole infrastructure is under attack.

In figure. 3 (b), the MPE graph for multiple flows (flows=5) remains intact whereas for the same number of attacked links, the graph for single flow declines rapidly. Multiple flow graph includes cumulative MPE for all of the flows, even if one degrades rapidly, if other flows have less degradation the overall MPE will decrease less rapidly.

5.4 Delay Overhead Evaluation of EPRM

As EPRM may result in a virtual path with increased hops as compared to the actual physical path. Therefore, we also conduct experiments to test the delay introduced by EPRM when it uses 2 peers to increase the defense against the DDoS attack. We performed three experiments using the topology from Fig. 1 where source sends `ping` commands to the destination. In our first experiment, we configure the network such that the `ping` command follows the static physical path. In second experiment, we configure the system to follow the path $\langle \text{source}, p_1, p_3, \text{destination} \rangle$, and in third experiment we install ToR implementation on source node and ping the end-host/destination in 1, via anonymous ToR routers. The results of the 2,000 measured delay samples and their Cumulative Distributed Function (CDF) are illustrated in Fig. 4(a), and 4(b). These results show that by adding even two peers in the path, the increase in delay for EPRM is minimal, i.e., it increases from average 70 milliseconds to only 85 milliseconds and we do not experience any significant variations. We observed minimum to no difference when we increased the peers from 2 to 3. Whereas the delay overhead for ToR implementation is significantly larger than the overhead of static path (2-3 fold), while it only preserves anonymity. Whereas EPRM defends against infrastructure level attacks while adding minimal overhead. These results clearly demonstrate the suitability of EPRM for critical communications.

5.5 Overhead of SMT formalization for E2E Reachability

We have used simulation based experiments for the rest of the evaluation using large network topologies. We claim that the proposed formalism for reachability is efficient then previously proposed approaches [3, 4]. We also compare our approach with Dijkstra's algorithm [1], the most efficient algorithm available for shortest path calculation. Fig. 5 (a) shows the time required by SMT for finding set of potential neighbors/peers (a single logical path with l=*3*), which a source can contact for help if under attack. We can see that the time required by SMT increases when the network size increases, but E2E reachability formalization is still efficient than Dijkstra's algorithm. Note that E2E reachability formalization doesn't not incorporate the QoS or security constraints, and only finds a satisfiable solution for reachability, a path/route through which a source can reach destination under divers situation.

5.6 Evaluation of Off-line Phase

Fig. 5 (b) shows the time required by SMT for E2E reachability with QoS constrains. The time overhead in this case increases many fold, as compared to the E2E reachability without *QoS Constraints*, due to the large number of constraints (as SMT has to explore more choices to find a satisfiable solutions). Although the overhead increases with the networks size, but it is still feasible for practical life, as this

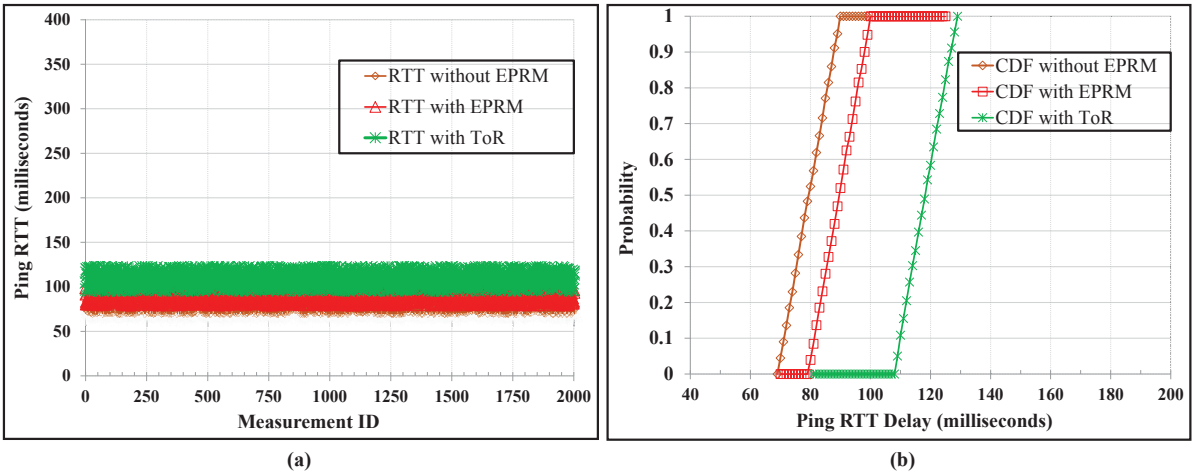

Figure 4: (a) Impact of EPRM on the bounded delay and its CDF analysis in (b)

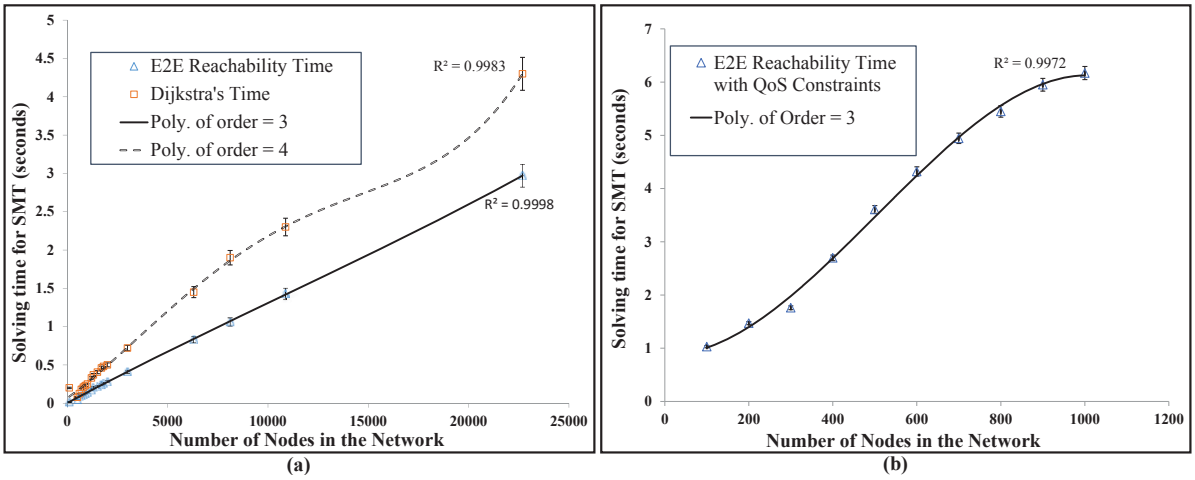

Figure 5: (a) Evaluation of resilient E2E reachability approach, (b) Time Required by SMT for E2E reachability with QoS constraints

computation is done as a part of off-line phase in EPRM protocol. Fig. 6 shows the time required to compute complete *Mutation Set*.

5.7 Evaluation of Online Phase

Maintaining the quality of service and defending an infrastructure efficiently depends on the deployed run-time defense mechanisms. All of the above steps of the protocol are suppose to run off-line repetitively, after certain period of time. Therefore, little lag or overhead is acceptable, but the run-time defense mechanisms should be much more efficient. We compare the efficiency of online phase of our approach with recent approach in MTD domain [3]. Due to the huge scale differences we separately present RRM graphs.

Figure 7 (a) shows the SMT solving time for route selection using RRM in a network that have 5 consecutive flows, whereas, w is the number of intervals that the new route should not repeat [3]. We can see that the required time increases when the network size increases, especially when the number of routers in the network reaches 300.

We use same statistics to evaluate our approach. We choose 5 nodes as source at the same time to transmit their flows, and use intermediate routers and peers to forward their traffic to the desired unique destinations. We repeat experiment 10 times and calculate the average time taken by a source for complete and distinct route selection. Note that we do not rely on human selection to configure manually and select for what period of time a route should not repeat, rather we rely on our algorithm to minimize the route intersection, and evenly distribute traffic on all links. As algorithm can efficiently find the mutation cycle, therefore we do not rely on manual selection. Figure 7 (b) shows the efficiency and overhead of online phase for the proposed approach. We calculate the duration for *Mutation Cycle* computation for different size of networks and in each network the cardinality of MS is five at most, which means MS contains at most five virtual paths. Note that the aim of online-phase of the protocol is not to just select a virtual path (set of intermediate peers), rather to select and mutate the possible sequences of peer in non-orderly fashion for different set of flows. Experimentation shows that the overhead of the proposed approach is significantly less than

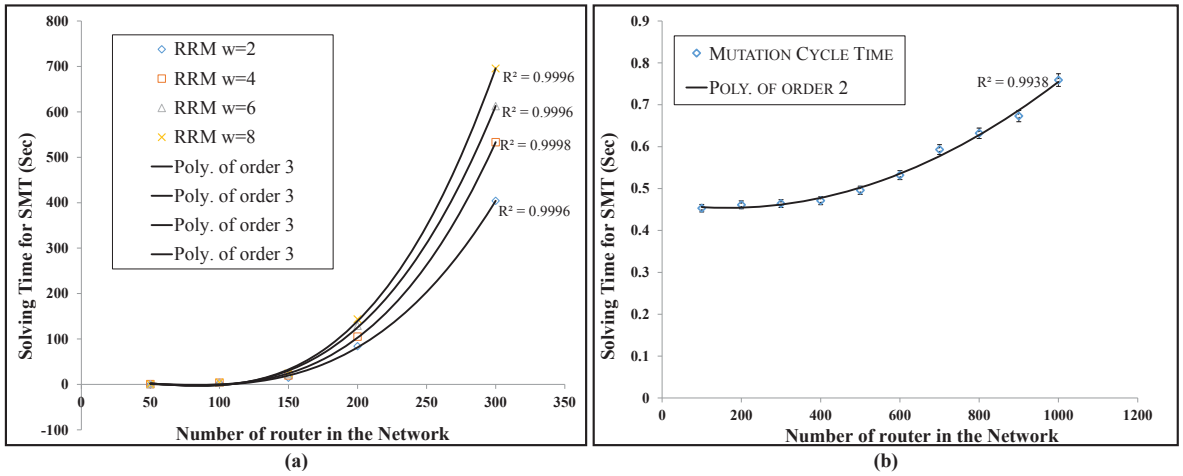

Figure 7: (a) Time required by RRM for Route selection, (b) Time required by SMT for on-line computation of Mutation Cycle

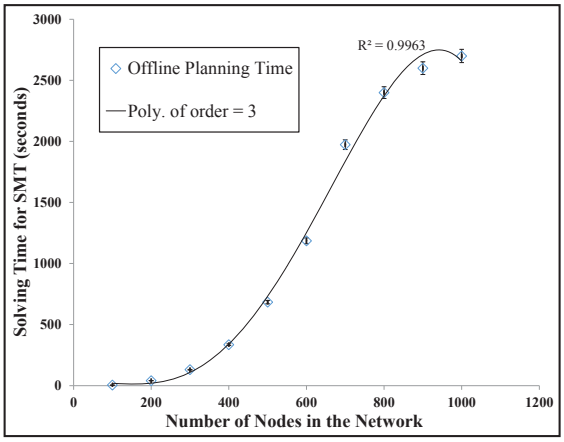

Figure 6: Time required by SMT for off-line computation of Candidate Mutation Set

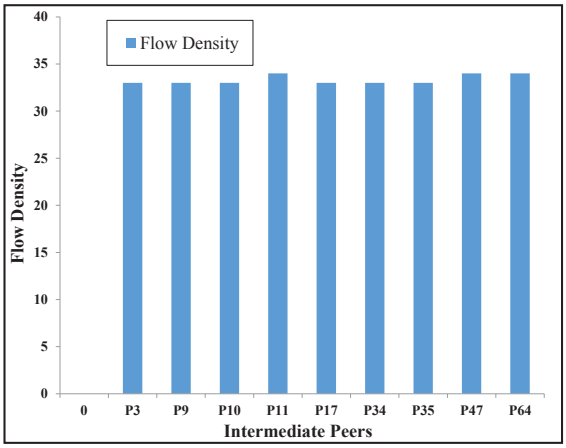

Figure 8: Flow density of utilized intermediate peers

RRM, which makes it suitable for practical usage, as mutation is performed in milliseconds as a result of the liveness response. The main reason for less overhead of our approach is that it is distributed in nature and all the computations are neither performed online nor by a single centralized controller. We also perform experiments to analyze whether the flow density is evenly distributed over the links, or not. The proposed approach aims to maximize the even distribution of flows, in order to defend the maximum percentage of flows and to deceive reconnaissance. Fig. 8 shows the distribution of flows over the peers and intermediate links. The network (with nodes=100 and edges=256) used for the investigation, only contains disjoint routes/paths for reaching destination.

6. CONCLUSION AND FUTURE WORK

In this paper, we addressed the limitations of existing route mutation approaches and proposed to extend route mutation architecture by considering end-hosts as routing elements. The resultant architecture exhibited significant advantages over existing approaches, in terms of run time efficiency, and resiliency against persistent attacks. The evaluation have showed that our algorithm is 40% more efficient than the Dijkstra's algorithm, 20% more efficient than RRM (in worst-case scenario) and more than 45% flows can be protected in multi-flow mode, even when the adversary is attacking 80% of the links. This makes EPRM more suitable for mission critical applications, when service availability and routing resilience for fraction of flows (such as emergence responding, communication in critical infrastructures, etc.) is the foremost objective. Ideally, our model if integrated with ToR infrastructure can address the issues of security/defense against DDoS attacks (by integrating adaptiveness) and can enhance the efficiency of ToR infrastructure.

In future we aim to: (1) incorporate additional operational constraints, (2) further analyze and evaluate the effectiveness of the end-point routing agility under adaptive attack models based on game theory, (3) deploy the proposed approach to other related architectures e.g. wireless ad-hoc networks, and (4) extend the existing approach to defend against reconnaissance attacks. We also notice absence of fairness for utilization of the resources in multi-flow cases which perhaps is the reason for rapid drop of MPE

when more then 50% links are attacked, therefore, in future (5) we also aim to exploit the potential of centralized SDN controller for fair distribution of resources (e.g. bandwidth) among peers. Finally, (6) we aim to develop and integrate quantitative metrics in our model to measure how exposed/vulnerable an organization (or end-host) is against modern DDoS attacks, and quantify that how much resilience our approach can offer to a certain end-host or an organization.

7. ACKNOWLEDGMENT

This research is based upon the work supported by the U.S Army Research Office (ARO) and Asymmetric Resilient Cybersecurity (ARC) initiative at PNNL (under Contract DE-AC05-76RL01830). Any findings, conclusions, or recommendations expressed in this research are those of author(s) and do not necessarily reflect the views of the sponsor.

8. REFERENCES

[1] E. Dijkstra, "A note on two problems in connexion with graphs," *Numerische Mathematik*, vol. 1, no. 1, pp. 269–271, 1959.

[2] F. Gillani, E. Al-shaer, S. Lo, Q. Duan, M. Ammar, and E. Zegura, "Agile virtualized infrastructure to proactively defend against cyber attacks," in *INFOCOM 2015*, vol. 1, pp. 270–280 vol.1 , April 2015.

[3] Q. Duan, E. Al-Shaer, and H. Jafarian, "Efficient random route mutation considering flow and network constraints," in *Communications and Network Security (CNS), 2013 IEEE Conference on*, Oct 2013.

[4] J. Jafarian, E. Al-Shaer, and Q. Duan, "Formal approach for route agility against persistent attackers," in *Computer Security, ESORICS 2013*, ser. Lecture Notes in Computer Science, J. Crampton, S. Jajodia, and K. Mayes, Eds. Springer Berlin Heidelberg, 2013.

[5] "PlanetLab," in *http://www.planet-lab.org*.

[6] "PlumGrid," in *http://www.plumgrid.com/*.

[7] M. Faloutsos, P. Faloutsos and C. Faloutsos, "Openflow random host mutation: On Power Law Relationships on the Internet Topology," in *In Proc. ACM SIGCOMM*, 1999.

[8] R. Dechter, *Constraint Processing*. San Francisco, CA, USA: Morgan Kaufmann Publishers Inc., 2003.

[9] F. Rossi, P. v. Beek, and T. Walsh, *Handbook of Constraint Programming (Foundations of Artificial Intelligence)*. New York, NY, USA: Elsevier Science Inc., 2006.

[10] L. De Moura and N. Bjørner, "Z3: An efficient smt solver," in *Proceedings of the Theory and Practice of Software, 14th International Conference on Tools and Algorithms for the Construction and Analysis of Systems*, ser. TACAS'08/ETAPS'08. Berlin, Heidelberg: Springer-Verlag, pp. 337–340, 2008.

[11] J. Leskovec and A. Krevl, "SNAP Datasets: Stanford large network dataset collection,"http://snap.stanford.edu/data, Jun. 2014.

[12] M. Davis and H. Putnam, "A computing procedure for quantification theory," *J. ACM*, vol. 7, no. 3, pp. 201–215, Jul. 1960.

[13] M. Fränzle, C. Herde, T. Teige, S. Ratschan, and T. Schubert, "Efficient solving of large non-linear arithmetic constraint systems with complex boolean structure," *Journal on Satisfiability, Boolean Modeling and Computation*, vol. 1, pp. 209–236, 2007.

[14] M. R. Garey and D. S. Johnson, *Computers and Intractability; A Guide to the Theory of NP-Completeness*. New York, NY, USA: W. H. Freeman & Co., 1990.

[15] G. S. Kc, A. D. Keromytis, and V. Prevelakis, "Countering code-injection attacks with instruction-set randomization," in *Proceedings of the 10th ACM Conference on Computer and Communications Security*, ser. CCS '03, New York, NY, USA: ACM, pp. 272–280, 2003.

[16] A. Medina, I. Matta, and J. Byers, "Brite: A flexible generator of internet topologies," Boston, MA, USA, Tech. Rep., 2000.

[17] H. Shacham, M. Page, B. Pfaff, E.-J. Goh, N. Modadugu, and D. Boneh, "On the effectiveness of address-space randomization," in *Proceedings of the 11th ACM Conference on Computer and Communications Security*, ser. CCS '04. New York, NY, USA: ACM, pp. 298–307, 2004.

[18] A. Shamir, "How to share a secret," *Commun. ACM*, vol. 22, no. 11, pp. 612–613, Nov. 1979.

[19] T. Shu, M. Krunz, and S. Liu, "Secure data collection in wireless sensor networks using randomized dispersive routes," *IEEE Transactions on Mobile Computing*, vol. 9, no. 7, pp. 941–954, July 2010.

[20] Jia, Quan and Wang, Huangxin and Fleck, Dan and Li, Fei and Stavrou, Angelos and Powell, Walter, "Catch me if you can: A cloud-enabled ddos defense", in *44th Annual IEEE/IFIP International Conference on Dependable Systems and Networks*, pp. 264-275, 2014.

[21] Wood, Paul and Gutierrez, Christopher and Bagchi, Saurabh, "Denial of Service Elusion (DoSE): Keeping Clients Connected for Less", in 34th IEEE Symposium on Reliable Distributed Systems (SRDS), 2015.

[22] Z. Ye, S. Krishnamurthy, and S. Tripathi, "A framework for reliable routing in mobile ad hoc networks," in *INFOCOM 2003.*, pp. 270–280, March 2003.

[23] M. S. Kang, S. B. Lee, and V. D. Gligor, "The crossfire attack," in *Proceedings of the 2013 IEEE Symposium on Security and Privacy*, ser. SP '13. Washington, DC, USA: IEEE Computer Society, pp. 127–141, 2013.

[24] D. B. Johnson, D. A. Maltz, and J. Broch, "Ad hoc networking." Boston, MA, USA: Addison-Wesley Longman Publishing Co., Inc., ch. DSR: The Dynamic Source Routing Protocol for Multihop Wireless Ad Hoc Networks, pp. 139–172, 2001.

[25] "NEPI," in *http://nepi.inria.fr/*.

[26] S. Bhatia, S. Di Giovanni, T. Haddow, A. Bavier, S. Muir, and L. Peterson, "Vsys: A Programmable sudo," in *USENIX Annual Technical Conference*, 2011.

[27] A. Studer and A. Perrig, *The Coremelt Attack*. Berlin, Heidelberg: Springer Berlin Heidelberg, pp. 37–52, 2009.

[28] D. M. Goldschlag, M. G. Reed, and P. F. Syverson, "Hiding routing information," in *Proceedings of the First International Workshop on Information Hiding*. London, UK, UK: Springer-Verlag, pp. 137–150, 1996.