# Sanctuary Trail: Refuge from Internet DDoS Entrapment

Hsu-Chun Hsiao, Tiffany Hyun-Jin Kim, Sangjae Yoo, Xin Zhang,

Soo Bum Lee, Virgil Gligor, and Adrian Perrig

June 7, 2012

CMU-CyLab-12-013

CyLab
Carnegie Mellon University
Pittsburgh, PA 15213

# Sanctuary Trail: Refuge from Internet DDoS Entrapment

Hsu-Chun Hsiao[†], Tiffany Hyun-Jin Kim[†], Sangjae Yoo[†], Xin Zhang[§]
Soo Bum Lee[†], Virgil Gligor[†] and Adrian Perrig[†]
[†]CyLab / Carnegie Mellon University [§]Google

**Abstract.** We propose STRIDE, a new Internet architecture that provides strong DDoS defense mechanisms for both public services and private end-to-end communication. This new architecture presents several novel concepts including long-term static paths, bandwidth allocation through a top-down topology discovery protocol, dynamic bandwidth allocation via network capabilities, and differentiated packet prioritization. In concert, these mechanisms provide 1) a strong static-class bandwidth guarantee, 2) strongly guaranteed capability establishment for private end-to-end communication, and a linear waiting time guarantee in the number of malicious source domains for capability establishment for public services, and 3) globally fair bandwidth allocation for capability-protected flows. STRIDE addresses the denial-of-capability problem and defends against a Coremelt attack by preventing a botnet from crowding out other flows on bottleneck network links. We demonstrate these properties through formal analysis and simulation.

## 1. INTRODUCTION

An important research question is how to construct an architecture that provides strong defenses against DDoS attacks; i.e., what fundamental architectural primitives will support effective DDoS defenses? Since DDoS attacks are fortunately the exception rather than the norm, a desired property is that the architecture should enable efficient operation also during peace time. A viable architecture should also support different communication modalities, namely, public services such as Google, CNN, etc, and private end-to-end communication where the destination knows in advance the potential senders. In an effective DDoS defense architecture, a legitimate client can reach a public server despite a massive flooding attack by a large-scale botnet. In the case of private end-to-end communication between two end-hosts, they should be able to communicate despite botnet flooding attacks that know their network locations.

Unfortunately, the currently deployed DDoS defenses are a far cry from these availability properties, as a botnet can easily prevent communication. In fact, a recent world-wide security survey [1] suggests that Botnet-driven DDoS attacks have gone mainstream as a low cost, high-profile form of cyber-protest. Both the attack intensity and frequency have drastically accelerated in the past few years: the largest reported attack size doubled every year, to more than 100Gbps seen in 2010. The majority of network operators in this survey also ranked DDoS attacks as the biggest threat.

While researchers have studied numerous DDoS defenses over the past decade [6, 11, 14, 15, 20, 27, 30, 36], they have worked under the constraint to provide solutions that are applicable to the *current* Internet. However, in this context an important research question is to consider how good DDoS defenses could be in a clean-slate world. An answer to this question will not only provide us with a blueprint for a highly available network architecture, but also reveal the chasm between the best current Internet approach and the best clean-slate approach, providing a measure of progress and what the benefits of a clean-slate deployment could be.

An effective test for strong DDoS defense is resilience against the Coremelt attack [29]. This attack has not been effectively addressed by any system to date. In a Coremelt attack, an adversary uses a large-scale botnet whose bots communicate only with each other, such that they overload intermediate network links. Since all inter-bot traffic is desired by the end-points, all the defenses that attempt to eliminate undesired traffic are ineffective. Ideally, a new network architecture would be resilient against this newly emerging threat.

To achieve strong DDoS defenses, we argue that a DDoS-resilient Internet architecture should inherently provide (i) inbound traffic control to Administrative Domains (ADs), (ii) regulated routing advertisement flow that facilitates bandwidth allocation, (iii) multi-path diversity and outbound path control, and (iv) real-time path congestion information to the endpoint ADs. Our goal in this paper is to explore and demonstrate how the new network architecture achieves DDoS resilience with these properties. Correspondingly, we propose a DDoS-resilient next-generation Internet architec-

ture, called STRIDE, which provides precise and differential *lower-bound* bandwidth guarantees for each authorized AD-level path, or the "sanctuary trails" that isolate attack traffic from legitimate communication. By leveraging network capabilities, an end-to-end flow can acquire a bandwidth share within a given time period if all the ADs on the path agree to supply that amount of bandwidth. STRIDE efficiently protects the first capability request packet, addressing the Denial-of-Capability (DoC) problem. In concert, STRIDE provides guaranteed bandwidth for private end-to-end communication and achieves linear waiting time in the number of malicious source domains for the first capability request packet. Finally, STRIDE achieves the above guarantees and properties *without* requiring per-path or per-AD state at routers in the fastpath.[1]

**Contributions.** We emphasize the need of providing intrinsic DDoS resilience in the network design, and identify the architectural properties that facilitate DDoS defense from the ground up. We propose a network architecture, STRIDE, that provides precise bandwidth guarantees for legitimate communication even in the presence of large-scale DDoS attacks; the new architecture addresses several open challenges such as the Denial-of-Capability attack and the Coremelt attack. Despite the strong DDoS resilience, STRIDE routers maintain no per-path or per-flow state in the fastpath and require no key establishment across administrative domains.

## 2. BACKGROUND

This paper presents a new DDoS-resilient architecture with strong defense guarantees. Before we elaborate on the novel concepts that STRIDE proposes, we first review the existing concepts that STRIDE leverages as a starting point: trust domains, endpoint-driven path selection, and network capabilities.

### 2.1 Trust Domains

The notion of Trust Domains (TDs) is first introduced in SCION [37], a clean-slate design of a secure Internet architecture. SCION divides ADs in the Internet into several TDs, where a TD is defined as "a set of ADs that agree on a coherent root of trust and have mutual accountability and enforceability for route computation under a common regulatory framework." Each TD contains a *TD core* (or TDC for short) consisting of the tier-1 ISPs, and every AD learns a set of half paths to reach its TDC (described below). As a result, end-to-end communication is achieved by splicing the source's and destination's half paths. Note that SCION's TD division can naturally confine attacks within a TD as the legitimate source's and destination's TDs do not route their traffic through an attacker's TD. Figure 1 illustrates the TD.

**Top-down topology discovery.** Within each TD, ADs discover paths to their TDC as follows. A TDC periodically broadcasts *path construction beacons* (PCBs) which contain *one-hop* paths starting from the core to its one-hop customer ADs. Upon receiving a PCB, an intermediate AD updates the PCB for each of its downstream ADs (e.g., customers and peers) with the local topology as follows: the $i^{th}$ intermediate AD ($\text{AD}_i$) along a path $\pi$ appends the ingress and egress interfaces for a particular downstream AD ($\text{AD}_{i+1}$) on path $\pi$ to the PCB. That is:

$$I_i^\pi = ingress_i^\pi \| egress_i^\pi \| AD_{i+1}^\pi, \tag{1}$$

where $ingress_i^\pi$ and $egress_i^\pi$ stand for the ingress and egress interfaces of $\text{AD}_i$, and $\text{AD}_{i+1}^\pi$ represents the downstream AD on path $\pi$ to receive this PCB. Then $\text{AD}_i$ propagates the updated PCB to the designated downstream AD ($\text{AD}_{i+1}$). $\text{AD}_i$ repeats this process for other downstream ADs on different paths. As a result, intermediate ADs can learn the path information to reach the TDC.

PCBs also contain digital signatures and cryptographic authenticators called *opaque fields* to protect the integrity of the routing information. During PCB propagation, $\text{AD}_i$ on path $\pi$ constructs an opaque field $O_i^\pi$ as follows:

$$\begin{aligned} O_i^\pi &= I_i^\pi \| M_i^\pi, \\ M_i^\pi &= \text{MAC}_{K_i}(I_i^\pi \| M_{i-1}^\pi), \end{aligned} \tag{2}$$

where the short cryptographic Message Authentication Code (MAC) is computed using a secret key $K_i$ known only to $\text{AD}_i$. The secret key $K_i$ is derived based on the expiration time $t_i$ of this link, e.g., $K_i = f(t_i, K_i^{master})$, where $f(\cdot)$ is a pseudo-random function and $K_i^{master}$ is $\text{AD}_i$'s master secret key.

Hence, the PCB that $\text{AD}_i$ propagates to the designated downstream AD ($\text{AD}_{i+1}$) on path $\pi$ is:

$$\begin{aligned} PCB_0^\pi &= timestamp \| O_0^\pi, \\ PCB_i^\pi &= PCB_{i-1}^\pi \| O_i^\pi \| t_i \| signature_i, \end{aligned} \tag{3}$$

where $signature_i$ represents the digital signature on this PCB using $\text{AD}_i$'s private key and *timestamp* represents the expiration time of path $\pi$.

$\text{AD}_i$ repeats this process for other downstream ADs on different paths, and upon receiving PCBs, intermedi-

[1]Fastpath and Slowpath refer to different processing elements that handle packet forwarding in a router. The Fastpath refers to packet processing by dedicated hardware, for example, by the linecard in a router. Slowpath refers to packet processing by a general CPU, which often results in a packet processing latency that is 2 magnitudes slower than the Fastpath.

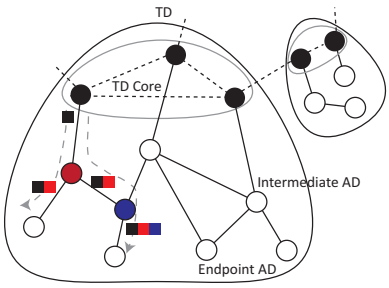


Figure 1: Trust domain example. Each node represents an AD, and the three black nodes represent the tier-1 ADs that constitute the TDC.

ate ADs learn the path information to reach the TDC.[2]

Opaque fields enable packet-carried state forwarding, which requires no routing state at routers. To send data packets along $\pi$, the sender embeds in the packet header opaque fields $\{O_i^\pi | AD_i \text{ on } \pi\}$, each of which remind an intermediate AD of it own decision that $\pi$ is an approved path for carrying data packets.

## 2.2 Endpoint-Driven Path Selection

Due to the topological relationships among ADs, an endpoint AD is likely to receive multiple PCBs with different path information.

Among all the paths that an endpoint AD learns from PCBs, the endpoint AD selects (up-to) $k$ up-paths for reaching the TDC and (up-to) $k$ down-paths for receiving packets from the TDC, where up-paths and down-paths may overlap.

**Path server.** Each endpoint AD uploads its down-paths to the *path server* in the core. To construct a complete end-to-end communication path for a destination, a source AD ($AD_s$) first uses its up-paths to reach the path server in the TDC which returns the destination's down-paths to $AD_s$. $AD_S$ then selects one of the returned down-paths to construct an end-to-end path.

**Shortcuts.** Before naively combining one of the source AD's up-paths with one of the destination AD's down-paths, the source AD searches the paths for peering links to find a "shortcut" path without going through the TDC. We discuss how STRIDE performs with shortcuts in Appendix B.1.

## 2.3 Network Capabilities

Network capabilities [33, 36] are widely used in prior DDoS defense systems to enable destination-controllable flow prioritization and stateless filtering at intermediate routers. To construct a capability, a sender sends a capability request to the destination, and every router on the forward path adds to the packet header a cryptographic authenticator that can only be computed and verified by that router. If the destination grants the request, it returns all the authenticators (representing the capability) to the sender. In general, packets with capabilities have precedence over packets without capabilities.

The authenticator can be implemented in many ways. Since we focus on an AD-level architecture, we compute per-AD authenticators, rather than per-router in a way similar to the opaque field construction, which preserves path integrity while giving ADs flexibility in intra-domain routing. For example, MACs in opaque fields can represent a network capability of the associated path.

[2]We assume that links are bidirectional.

# 3. PROBLEM DEFINITION

## 3.1 Threat Model

We consider massive Distributed Denial of Service (DDoS) attacks launched by a botnet (which consists of a large number of malware-infected machines; i.e., bots). In DDoS attacks, bots are orchestrated to send excessive data traffic to exhaust the bandwidth of network link. In particular, we address two types of DDoS attacks, namely, 1) disabling connection setup in capability-based protocols (denial-of-capability (DoC) attack [7]), and 2) exhausting the bandwidth of established connections. Both attacks can be launched against network infrastructure or against specific servers. In bandwidth exhaustion attacks, we especially focus on the Coremelt attack [29] that aims to overload a target ISP's backbone network using a large number of legitimate-looking flows established among colluding bots (hence, any attempt to identify the attack based on flow's bandwidth would fail). Both DoC and Coremelt attacks are considered the most powerful DDoS attacks that prevent end-to-end bandwidth guarantees, to date.

## 3.2 Desired Properties

We aim to achieve the following properties for a DDoS-resilient network architecture.

**Precise bandwidth guarantees.** Precise end-to-end bandwidth guarantees should be provided to legitimate communication flows even in the presence of a large-scale DDoS attack. Furthermore, the effect of attacks should be confined to infested domains.

**Robustness and efficiency.** Network elements should be resilient to DDoS attacks. This essentially requires efficient protocols and network devices.

**Informed path control at the edge.** Endpoint ADs should be notified of the network congestion status in real-time. On identifying congestion, endpoint ADs should be able to reroute traffic to avoid congestion: a source AD needs to have multiple paths to reach a destination; a destination AD should be able to (1) hide/disclose paths for private/public communication, and (2) change inbound traffic paths to shift traffic.

## 3.3 Assumptions

In designing a new DDoS resilient Internet architecture, we only make three fundamental assumptions that can be justified using existing security mechanisms. We do not make additional assumptions that would broaden the attack surface.

**TDC cannot be congested.** The TDC is highly provisioned such that flows cannot congest internal TDC links. Since the TDC can accurately assess its capacity requirement (bandwidth allocation is initiated by itself), it can provision its capacity to avoid congestion.

**Routers within the same TD are trusted.** We assume a botnet adversary in this paper that infects end-hosts. We consider that routers are well administered and uncompromised. In Section 7.2, we discuss the impact of compromised routers.

**Path Server is available.** A successful DDoS attack against Path Servers would disable end-to-end path establishment in STRIDE. Such an attack can be prevented in two ways: (1) TDC can detect the origin of attack traffic against Path Servers and throttle their traffic, and (2) the bandwidth-guaranteed paths STRIDE offers can be used to contact a Path Server.

## 4. STRIDE: DESIGN OVERVIEW

In this section, we sketch how our new architecture, STRIDE, provides guaranteed end-to-end data delivery to legitimate flows even in the presence of DDoS attacks. The STRIDE architecture uses several mechanisms in conjunction to provide these properties.

**Bandwidth classes.** Link bandwidth is split up into differentiated bandwidth classes: static, dynamic, and best effort. Static bandwidth is for long-lived but low-bandwidth guarantees, while dynamic is for short-lived but high-bandwidth guarantees. Best effort can use up the unused bandwidth through statistical multiplexing.

**Bandwidth allocation.** Static and dynamic bandwidth is allocated for half-paths that are created during Path Construction Beacon (PCB) propagation.

**Half-path bandwidth guarantees.** A half-path received through a PCB is a best-effort path, offering no guarantees. However, an AD can *activate* up to $k$ best-effort half-paths to convert each into a static half-path, offering a guaranteed amount of bandwidth from that AD to the TDC.

**Best-effort and static channels.** A channel is a conduit to carry traffic from a source to a destination. A single half-path can be used as a channel to reach the TDC. Two half-paths can be combined to form an end-to-end channel between a source and a destination – we call the half-path of the source the up-path (as traffic on that channel traverses ADs upwards towards the TDC) and the half-path of the destination the down-path. If two best-effort half paths are combined, the resulting channel is a best-effort channel, and similarly, two static half-paths create a static channel. Hybrid channels are also possible, and we will make use of a static up-path that is combined with a best-effort down-path.

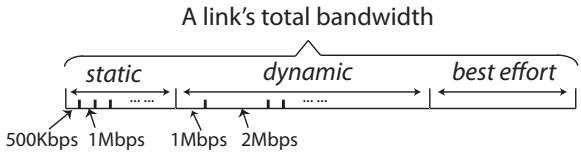


Figure 2: Three traffic classes of STRIDE.

**Dynamic channel.** Any channel can be converted into a dynamic channel, in order to reserve a high-bandwidth channel for a short amount of time (on the order of seconds to enable fast revocation). All ADs, including the destination, need to agree on the amount of bandwidth offered for that channel. This channel is similar to network capabilities previously discussed.

**Best effort traffic priorities.** Packets flowing through a best-effort channel have several levels of priority, depending on whether they initially flowed up over a static half-path in case of a hybrid channel (and whether the packet exceeded the static capacity), or whether the packets traversed a congested link. Best-effort traffic from within the TDC also obtains priority over external best-effort traffic. This approach will help throttle bots that are attempting to mount a DoC attack, as a legitimate host's packets traversing its own static up-path with a best-effort down-path towards the congested public server will obtain a high priority.

**Dynamic channel priorities.** The allocation of dynamic bandwidth across several competing dynamic channels is accomplished through fair-sharing based on the dynamic channel bandwidth allocation of the PCB. While the packet travels up-path towards the TDC, the allocation of the source is used, and similarly the allocation of the destination domain is used on the down-path. The intuition here is that a large number of bots within a domain will compete for bandwidth amongst each other. This is one of the key insights that helps us defend against Coremelt attacks.

One surprising aspect of all these mechanisms is that they do not require any per-flow state in the fastpath for forwarded traffic; only the initial channel establishment requests require slow-path operations for flow admission. With these ingredients, we are going to construct a series of mechanisms that achieve the highly available communication we strive for. In a nutshell, static paths are kept secret and are used to provide guaranteed communication for private services. To establish communication with a public server that is under attack, a hybrid static-best-effort channel is used (as described above) to establish a dynamic channel, achieving both defense against DoC and Coremelt. Next, we provide additional detail of these mechanisms.

### 4.1 Three Traffic Classes

To support flexible bandwidth allocation, STRIDE defines three traffic classes that each AD assigns to its link as follows.

**Static class.** This traffic class is for guaranteed, persistent long-term lower-bound bandwidth that ADs allocate, for example to protect initial connection setup request packets between a source and a destination. Each AD divides static-class bandwidth into a number of shares (i.e., lower-bound guarantees) and assigns them to paths that traverse the AD. Each AD allocates a small portion of its total bandwidth (e.g., 5–15%) to this class.

**Dynamic class.** This traffic class is for guaranteed, short-term end-to-end bandwidth allocations, and supports high capacity channels. The dynamic class may account for the majority of the link capacity (e.g., 60–65%).

**Best-effort class.** Each AD allocates the remainder of its bandwidth (e.g., 30% of its total link capacity) for best-effort traffic. Best-effort traffic can take advantage of statistical multiplexing and take over unused bandwidth from the other two classes.

Figure 2 is an illustration of the bandwidth spectrum of a link in STRIDE. We provide guidelines on how an AD can divide its total link capacity to the above three classes in Appendix A.2.

Within the static or dynamic traffic class, an AD can assign different traffic sub-classes (e.g., 500Kbps, 1Mbps, etc) to individual paths/flows based on the empirically measured flow size distribution. For example, the fraction of dynamic bandwidth allocated to the 1Mbps sub-class can be derived based on the fraction of flows with 1Mbps rate in the current Internet.

## 4.2 Half-Path Setup

We now describe the steps of an endpoint AD to establish a guaranteed static path to its TDC.

① **Bandwidth announcement:** As a PCB travels from the TDC to endpoint ADs, each AD adds information regarding its own bandwidth allocation. In particular, an intermediate AD announces the amount of bandwidth that it can guarantee for each of its children for the static channel. Figure 3 depicts the bandwidth allocation (denoted by the numbers) of PCBs for the static channel. This particular diagram shows that $AD_S$ has three possible paths supporting static channels:

❶ $TDC \overset{100}{\rightarrow} AD_A \overset{80}{\rightarrow} AD_F \overset{40}{\rightarrow} AD_S$
❷ $TDC \overset{100}{\rightarrow} AD_B \overset{50}{\rightarrow} AD_F \overset{25}{\rightarrow} AD_S$
❸ $TDC \overset{100}{\rightarrow} AD_B \overset{50}{\rightarrow} AD_E \overset{25}{\rightarrow} AD_S$

In addition, the opaque fields in a PCB contain a *best-effort capability* by which the endpoint AD can communicate with the TDC on the best-effort channel.

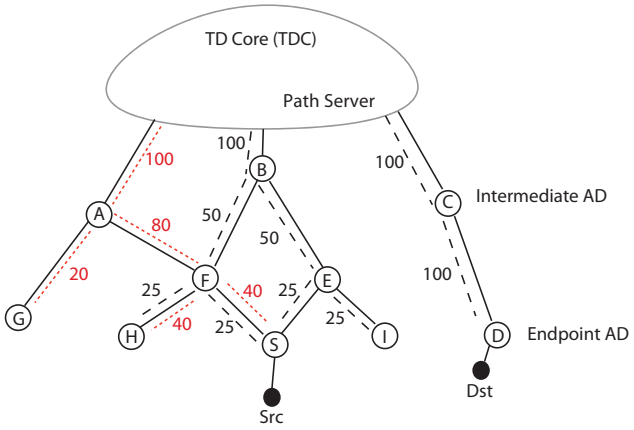


Figure 3: Illustration of STRIDE and how bandwidth is allocated on PCBs.

② **Activation:** Upon receiving a PCB with the bandwidth allocation information, an endpoint AD ($AD_s$) sends an activation request to the TD Core ($TDC$) to reserve the static bandwidth allocated in the PCBs. While forwarding this request, all the intermediate ADs allocate the requested static bandwidth for $AD_s$ if they can support the request. For example, $AD_S$ may request 40 units of static-class bandwidth through path ❶, 25 units through path ❷, etc.

③ **Confirmation:** The $TDC$ sends a confirmation message to $AD_s$ for the guaranteed static channel that $AD_s$ (and all of its hosts) can use to reach $TDC$. The new opaque fields constructed along the confirmation contain a *static capability* by which the endpoint AD can communicate with the $TDC$ on the static channel. Note that STRIDE allows each endpoint AD to create up to $k$ half-paths per its upstream ADs, and each endpoint AD has the freedom to keep a subset of half-paths in private for privileged access, as will be explained in ④, while registering others at the Path Server for public usage by any sources.

## 4.3 End-to-end Channel Setup

We now describe how an up-path and a down-path can be combined to establish a channel.

④ **Path selection:** When a source endhost $Src$ of $AD_s$ wants to communicate with a destination endhost $Dst$ of $AD_D$, $Src$ queries the Path Server of its TDC for the down-path to reach $Dst$. The Path Server then returns unconcealed down-paths to $Src$. Alternatively, $Dst$ can inform $Src$ of a private path over an Out-Of-Band (OOB) channel. By selecting one of the down-paths provided by the Path Server, $Src$ now establishes an end-to-end path for sending dynamic channel setup request.

⑤ **Dynamic channel setup request:** To acquire guaranteed bandwidth along a dynamic channel, $Src$ sends a dynamic channel setup request on this newly-established end-to-end path. While this request travels toward $Dst$,

all ADs on the path specify the bandwidth that they can provide for the dynamic channel and forward it toward $Dst$. STRIDE allows endhosts to send more traffic than what is allocated for enhanced utilization of available bandwidth (refer to Section 5.1). In case $Src$ sends requests beyond the allocated bandwidth of the static channel along the up-path, the endpoint AD ($AD_S$) indicates that the extra packets are beyond permitted allocation (i.e., $AD_S$ sets an overuse bit on each extra packet).

In case $Src$ cannot send the request using the static channel of the end-to-end path, maybe due to congestion on any of the announced down-paths, STRIDE flexibly allows the endpoint AD to send requests on the best-effort channel. Specifically, by including capabilities (best-effort or static) for the up-path and down-path, the request can travel through the static channel or best-effort channel for both the up- and down-paths, the static channel along the up-path and the best-effort channel along the down-path, or the other way around.

⑥ **Bandwidth allocation for the dynamic channel:** When $Dst$ successfully receives $Src$'s dynamic channel setup request, $Dst$ can deduce sustainable dynamic-class bandwidth for $Src$ based on the reported dynamic-class bandwidth availability of all the intermediate ADs (e.g., minimum of the dynamic-class bandwidth allocations of all the intermediate ADs). Then $Dst$ sends this information to $Src$, during which the *dynamic capability* is constructed along the return path. STRIDE provides flexible options for sending the reply. For example, $Dst$ may send the reply through the allocated dynamic channel or through any of the combinations of the best-effort and static channels.

⑦ **Guaranteed data transmission:** When $Src$ receives the reply, it can enjoy sending data traffic using the dedicated dynamic-class bandwidth by embedding the dynamic capability in the data packets. Since this bandwidth is short-lived, $Src$ may renew this dynamic-class bandwidth using actual data packets. Similar to step ⑤, $Src$ can also send more than permitted dynamic bandwidth allocation, in which case $AD_s$ sets the overuse bit for the extra data traffic.

# 5. STRIDE PROTOCOL DESCRIPTION

In this section, we elaborate STRIDE's path establishment mechanisms highlighted in Section 4; we first present the procedure of setting up best-effort (BE) and static half paths and then illustrate how endhosts can acquire end-to-end dynamic paths.

In Appendix A, we show that STRIDE provides a connection setup guarantee for both public and private services. Given that the first packet is secured, STRIDE further achieves a strong bandwidth guarantee for legitimate flows.

A *STRIDE packet header* contains a traffic class, a bandwidth allocation (for static and dynamic classes only), a timestamp when the path was created, and opaque fields, which contain forwarding information and cryptographic MACs (i.e., a capability) for the authenticity check (see Section 2 for a detailed description of opaque fields).

## 5.1 Long-Term Guaranteed Static-Path Setup

In STRIDE, the construction of static paths consists of three steps: path announcement, activation request, and confirmation. These steps enable ADs on the packet forwarding paths to reserve precise bandwidth guarantees and enable endpoint ADs to confine/avoid the effects of DDoS attacks using agile path control. Each step is described below.

### ① *Bandwidth announcement along PCBs*

Upon receiving a PCB, an intermediate AD forwards to its downstream AD after appending the bandwidth information, including (1) currently available and (2) reservable static-class bandwidth that the intermediate AD can guarantee to its downstream AD. When a PCB reaches an endpoint AD, it contains a *path* $\pi$ from the TDC to the endpoint AD and a bandwidth guarantee $G^{\pi}$ that can be provided once this path is activated (details in step ②). Other bandwidth usage information on this traversing PCB includes (3) currently available and (4) reservable dynamic-class bandwidth, (5) currently available best-effort bandwidth. This information enables downstream ADs and endhosts to deduce congestion status and make informed decision in selecting paths.

On each PCB, an intermediate $\text{AD}_i$ constructs an opaque field $O_i$ as described in Eq. 2 in Sec. 2 except that now the MAC secret key $K_i$ depends on the traffic class (i.e., the BE class) as well as the expiration time $t$ of this best-effort channel. This construction prevents malicious senders from accessing an unauthorized bandwidth class or from using expired paths. For example, $K_i = f(BE, t, K_i^{master})$, where $f(\cdot)$ is a pseudo-random function and $K_i^{master}$ is $\text{AD}_i$'s master secret key. The resulting opaque fields in the PCB contains a forwarding path $\pi$ and a *best-effort capability* that authorizes access to the best-effort channel on $\pi$.

**Path diversity vs. quality.** STRIDE enables an AD to allocate bandwidth for its customers, thanks to the top-down PCB announcements. Yet, bandwidth allocation is still challenging because of the tradeoff between path diversity (i.e., the number of different paths) and path quality (i.e., the allocated bandwidth to each path). An intermediate AD can offer higher path diversity by propagating a PCB to more children (or through more egress routers), enabling endpoint ADs to select paths. Yet, such path diversity reduces the bandwidth that can be allocated to each child since the bandwidth

contained in a PCB must be split for all children *recursively* as the PCB propagates to downstream ADs. To address this issue, we propose a bandwidth overbooking technique to enhance path diversity and quality simultaneously.

**Bandwidth overbooking.** Note that an endpoint AD can choose up to $k$ paths[3] out of all announced paths. This indicates that not all announced paths will be activated. As a result, any intermediate AD can announce greater reservable bandwidth (i.e., overbook) to their downstream ADs than the actual link capacity, and defer the actual bandwidth reservation later during the activation step (②). With such overbooking, denial of path activation could potentially be frequent if intermediate ADs substantially overbook their links. To address this issue, an intermediate AD overbooks its bandwidth such that the probability of bandwidth over-booking to its link is below a certain threshold, as we analyze in Appendix C.

#### ② *Activation*

An endpoint AD requests for a path activation along the reverse path of $\pi$ to the TDC. Each request consists of 1) information fields and opaque fields (extracted from the latest PCB) associated with the path, 2) a desirable expiration time of the path, and 3) desirable static-class bandwidth, which does not exceed the announced reservable bandwidth in the PCB. STRIDE considers an activation request as a control message, like PCBs, that is sent along the path to be activated using the opaque fields received during the path announcement as the capabilities.

For lightweight management and flexibility, the expiration time and bandwidth is chosen from a finite set of values. For example, the expiration time can be $t$ hours, where $t$ is an integer and $1 \leq t \leq T_{max}$.

Upon receiving an activation request, an intermediate AD, which has sufficient un-allocated bandwidth (i.e., spare capacity $\geq$ desirable bandwidth), temporarily allocates the requested bandwidth for this path, otherwise, the AD silently drops the request. Note that, as mentioned in Section 4.1, the intermediate ADs assign pre-defined sub-class bandwidth to each activation request for efficient bandwidth control. To minimize bandwidth waste, ADs recycle temporarily allocated bandwidth when the activation fails, which is indicated by the lack of a confirmation (step ③) before certain deadline (e.g., before the arrival of the next PCB). This bandwidth allocation and recycling can be efficiently performed with $O(T_{max})$ memory state.

Each endpoint AD is allowed to activate up to $k$ distinct paths per upstream AD (or provider). This policy is made in accordance with the observation that in the current Internet, large endpoint ADs often subscribe to multiple providers for increased capacity and path diversity. This $k$-path policy can be enforced either by the providers or the path server in the TDC.

[3] $k$ is determined by contract and is enforced by their provider.

#### ③ *Confirmation*

The TDC informs the endpoint AD a successful path registration by sending a confirmation message along the activated path $\pi$. The confirmation message also contains the expiration time $t$ and the allocated bandwidth $G_s^\pi$. The TDC also updates the path server to include this activated path. Upon receiving TDC's valid confirmation, each intermediate AD on $\pi$ converts the temporarily-allocated bandwidth to be long-term (until the expiration time).

Before forwarding the confirmation to the next hop, the AD updates its opaque field $O_i$ using a different MAC secret key $K_i$: $K_i = f(S, t, G_s^\pi, K_i^{master})$, where $S$ stands for the static class.

After receiving the confirmation of path $\pi$, the endpoint AD can forward packets on the static channel of $\pi$ by including the updated opaque fields (which contain $\pi$ and a *static capability* for using the static channel of $\pi$) in the header of the packets.

### 5.2 Securing End-to-End Dynamic Channel Requests

Establishing static paths provides *half path guarantees*, eliminating the continuous uncertainty in the current Internet. As a result, endpoint ADs learn multiple long-term, protected static paths to communicate with the TDC.

The remaining work includes combining an up-path (i.e., a path from an endpoint AD to the TDC) and a down-path (i.e., a path from the TDC to an endpoint AD) to form an end-to-end path for guaranteed dynamic channel setup between two endhosts (i.e., ensured delivery of the dynamic capability requests). With this end-to-end dynamic channel, STRIDE then provides bandwidth guarantees to short-term, high-bandwidth dynamic flows. We first delineate the end-to-end channel establishment within the same TD. We then elaborate how the endhosts in two different TDs can establish end-to-end channels.

#### ④ *End-to-End Path Selection*

Even though STRIDE ensures strong bandwidth guarantees for half paths, providing guaranteed connection setup between two endhosts is still challenging because ADs on the up-path cannot predict the status of the down-path (e.g., how many other up-paths or source ADs are using a down-path concurrently?). In this section, we describe how STRIDE ensures that endpoint ADs can securely establish end-to-end channels (within the same TD) such that endhosts can successfully send

dynamic capabilities with bandwidth guarantees.

**Combining up-path and down-path.** When an endhost $Src$ in $AD_s$ attempts to make a connection to another endhost $Dst$ in $AD_D$, $Src$ contacts $AD_S$ for path resolution. In return, $AD_S$ requests the Path Server in the TDC for a list of static paths to $Dst$, and the server returns valid paths. As a result, the $AD_S$ can select both an up-path (from itself to its TDC) and a down-path (from the TDC to $AD_D$).

**Path selection policy.** If $AD_S$ keeps on selecting paths based on the highest available bandwidth, it may end up selecting a single best path for all the source endhosts (besides $Src$), eventually congesting this path. To resolve this issue, endpoint ADs in STRIDE perform probabilistic path selection as follows: the endpoint AD selects a path with probability proportional to the path bandwidth guarantees. With this policy, the endpoint ADs are more likely to select uncongested paths and reduce the average number of trials. We evaluate a specific instance of this policy in Sec. 6.

**Private paths.** We introduce the notion of private paths, which endpoint ADs can use to provide guaranteed down-paths to preferred endhosts. In a nutshell, an endpoint AD keeps a subset of $k$ half paths as private and provides them to its destination endhosts such that they can selectively provide them to preferred source endhosts.

We define private services to be provided by those servers that can predict future customers (i.e., membership-based webmail servers, Amazon). A private server that only provides access to a closed community can provide guaranteed connection setup to community members by using member-only private down-paths as follows: a destination can selectively disclose its private down-paths to preferred sources. The private paths can be distributed via Out-Of-Band channels or by uploading encrypted private paths to a Path Server. As a result, a valued customer of Amazon, for example, can obtain a guaranteed static channel for sending capabilities to Amazon.

### ⑤ *Dynamic Channel Setup Request*

After step ④, a source endhost can send a dynamic channel setup request for guaranteed dynamic-bandwidth allocation. Along the end-to-end request path, each AD on the routing path $\pi$ computes an opaque field $O_i$ using a MAC key $K_i = f(D, t, B_i^\pi, K_i^{master})$, where $D$ stands for the dynamic traffic class, and $B_i^\pi$ is the dynamic-class bandwidth that $AS_i$ can dedicate for this connection. For efficient flow bandwidth control, ADs assign a pre-defined bandwidth sub-class to individual flows (e.g., multiples of 100 kpbs) instead of allocating bandwidth in a fine granularity (e.g., 95 kbps).

A request header carries two additional indicators that enable congested intermediate ADs to efficiently control link bandwidth:

- **Overuse bit:** A source AD sets an overuse bit of a request packet in case this source endhost is sending packets more than the reserved bandwidth of the static up-path channel.
- **Congestion bit:** Each AD sets a congestion bit when it experiences congestion in its static channel.

**Prioritization of requests.** When an AD receives more requests than what its outgoing links can afford, the AD has to discard some of the requests while maintaining the bandwidth guarantee provided by the static path. Depending on where the congestion occurs, we specify different techniques to prioritize requests while respecting the static-class bandwidth guarantees and ISP business relationships.

*1. Host contention at local ADs:* Static bandwidth contention can occur at endpoint ADs when they cannot support all the requests from their endhosts. To resolve this issue, a payment-based scheme can be adopted: each client informs its host AD how much it is willing to pay for using an static up-path to send a capability request, and the endpoint AD can arrange based on some objectives (e.g., maximize the AD's revenue).

*2. Congestion at TD Core's outgoing links:* Since the source host has no prior knowledge of how many up-paths want to use the selected down-path concurrently, congestion may happen when the down-path cannot provide sufficient bandwidth to all incoming requests. We can adopt a **proportional scaling** technique, where the TDC allocates currently-available bandwidth based on the up-path's bandwidth guarantee.

*3. Congestion at intermediate ADs:* An intermediate AD may process a request that either comes through a static path or a best-effort channel. Those ADs on down-paths may even need to process a request that comes from a different TDC. STRIDE assigns levels of priority to the capability requests such that those with lower priorities are dropped first in case of congestion. Here are the priority levels ordered from the highest to the lowest:

1. The request has been on a static channel and no overuse bit is set.
2. The request is coming through a best-effort channel and there is no congestion.
3. The request has been on a static channel and the overuse bit is set.
4. The request is coming through a best-effort channel and there is congestion.
5. The request is coming from a different Trust Domain (TD).

In case an intermediate AD still suffers from congestion, this indicates that some previous AD(s) on the path did not comply with the allocation policy. In this

case, the congested AS can turn on probabilistic monitoring [12] to identify overusing paths without setting the overuse bit, which results in router states linear to the number of overusing paths. The AD then throttles packets coming from those paths. In addition, because ADs in the same trust domain are within a common legal system, law enforcements can take legal actions against non-compliant ADs, which may deter ADs from misbehaving in the long run.

**External requests from other TDs.** External requests are restricted to using best-effort down-paths. To enforce this policy, TDCs discard external requests attempting to use static paths.

#### ⑥ *Dynamic-class bandwidth allocation.*

Through a capability request, a destination endhost can discover the bottleneck link(s) and the available dynamic-class bandwidth along the path. The destination constructs a reply packet, which carries 1) reserved dynamic-class bandwidth of this flow, 2) constructed opaque fields (which include a *dynamic capability*), and 3) expiration time which indicates the lifetime of the guaranteed dynamic bandwidth. The destination also indicates which AD-to-AD link(s) is the bottleneck for determining the bandwidth reservation.

As the packet travels back to the source, ADs update their dynamic bandwidth allocation and opaque fields to accurately reflect the available bandwidth and reduce the potential waste of bandwidth. In case the allocated end-to-end dynamic bandwidth does not meet the source's need (e.g., determined by an application service), the source may select an alternate path. Furthermore, in case the destination indicates that the bottleneck link is on the up-path, the source can make an informed decision to avoid the bottleneck link when selecting an alternate up-path.

**Capability Update.** A source can renew the short-term capability while communicating with the destination as follows: the sender sets a renewal bit in the header of the capability-protected dynamic-class packets. If the destination renews, the source AD invalidates the old capability (e.g., by keeping track of the latest capability for each flow and reject packets carrying old capabilities) to prevent misuse.

#### ⑦ *Guaranteed Data Transmission*

Upon receiving the reply of a dynamic capability (step ⑥), the sender can use the end-to-end dynamic channel for guaranteed data transmission. The sender can also flexibly choose other types of end-to-end channels for different guarantees, as summarized in Table 1.

**Regulation.** For per-flow bandwidth guarantees, endpoint ADs monitor per-flow data usage and regulate potential violation. For example, every endpoint AD ensures that the overuse bit is set in data packets whose flow rate exceeds the allocated value. Each intermediate AD is again responsible to police each outgoing link by dropping some of the overusing packets such that the dynamic-class bandwidth usage is no more than the allocated dynamic-class bandwidth for the link. In addition, intermediate ADs and the TDC can perform both real-time probabilistic monitoring and offline traffic analysis to identify misbehaving endpoint ADs that fail to regulate their clients. TDCs and ADs also monitor the per-TD bandwidth usage of dynamic-class traffic at each interface to isolate attack traffic from other TDs.

Table 1: Five end-to-end channels and their guarantees.

| | |
|---|---|
| static | guaranteed low-capacity throughput |
| static→BE | linear waiting time guarantee |
| BE→static | no guarantee |
| BE | no guarantee |
| dynamic | guaranteed high-capacity throughput |

## 6. EVALUATION

In this section, we evaluate STRIDE with respect to its effectiveness against DDoS attacks. We show the effectiveness of end-to-end bandwidth guarantees under large-scale attack scenarios. We also test the packet forwarding performance of STRIDE via real-field implementation by comparing the throughput of a STRIDE router with that of an IPv4 router.

***Simulation setup.*** For realistic simulation, we use a CAIDA AS-relationship dataset[4] to construct AD-level topology, and construct a single TD where a tier-1 AD that connects to 2164 endpoint ADs is chosen as the TDC. Though the AS-relationship dataset does not include all interface-level paths that STRIDE would construct, our analysis on the dataset reveals that AD-level path diversity is high enough to support STRIDE's path control and hence to evaluate STRIDE's path construction. Specifically, the endpoint ADs in the dataset have more than 40 different paths to the TDC on average as Figure 4 shows.

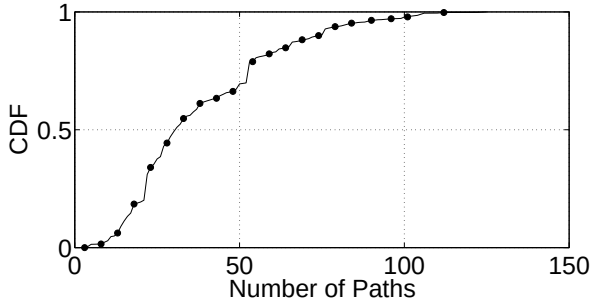


Figure 4: Number of paths at endpoint ADs.

[4]CAIDA. `http://as-rank.caida.org/data/`

If interface-level paths are constructed, path diversity at endpoint ADs would become much higher since the number of paths grows exponentially as PCBs propagate to downstream. For example, if the TDC and its immediate children have three different interfaces with their customer ADs, the number of paths available to endpoint ADs would grow nine times. We analyze the availability of bandwidth-guaranteed paths for our simulation topology and summarize the results in Appendix D.

***Bandwidth allocation.*** During the PCB propagation, each AD allocates bandwidth to its children ADs proportional to their sizes. We assume that the size of an AD is proportional to its degree and the minimum bandwidth allocation unit is determined by the number of paths (i.e., $k$) that an endpoint AD can activate. That is, under fixed physical link capacity and connectivity, large $k$ implies high path diversity with low bandwidth allocation.

## 6.1 Resilience against DoC Attacks

We evaluate the resilience of STRIDE against DoC attacks, under the following simulation scenario. We randomly select one hundred legitimate ADs (that are not contaminated by bots) and configure them to send traffic to a destination AD (randomly selected) using 10 different down-paths (i.e., $k$=10). Each source AD with a send rate equal to one tenth of the down-path capacity. Hence, in the absence of attacks, all down-paths are fully (but not overly) utilized. Then, we add contaminated ADs by bots (simply contaminated ADs) to the topology and set their send rate equal to that of a legitimate AD so as to make individual contaminated ADs indistinguishable from legitimate ones. The number of contaminated ADs is increased from 0 to 300. We note that even if the contaminated ADs send traffic to other destinations than the legitimate destination AD that we chose in this simulation, they would not affect the bandwidth of $k$ paths activated by the legitimate destination AD.

We evaluate the effectiveness of STRIDE against the above attacks in the following two scenarios.

**Public paths.** All legitimate and contaminated ADs use the $k$ activated down-paths to setup capabilities with the destination AD (i.e., they use the static channel).

**Private paths.** Half of the legitimate ADs use a non-public, private down-path to the destination that was provided to the source ADs in a secret out-of-band channel. Meanwhile, the remaining half of the legitimate ADs and contaminated ADs use the public paths as before.

As an evaluation metric, we use the *admission ratio*, which is defined as the percentage of the legitimate packets that successfully go through the bottleneck link/path.

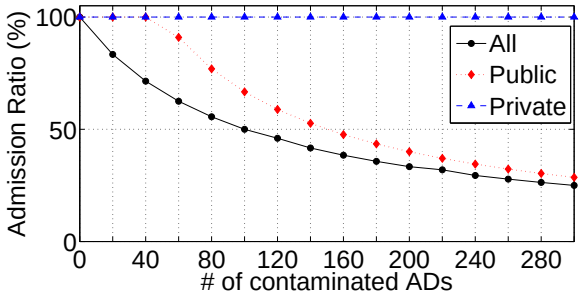


Figure 5: The admission ratio of legitimate packets with an increasing number of contaminated ADs. "All" represents the result when all legitimate packets traverse the public down-paths to the destination. Public and Private represent the results when half of the legitimate source ADs utilize the public paths and the other half utilize the private paths.

Figure 5 shows that when all source ADs use the public paths ("All"), the admission ratio of the legitimate packets decreases as more contaminated ADs are added since the per-AD bandwidth decreases. When half of the legitimate ADs acquire a private path from the destination ("Private") [5], their packets are unaffected by the attack as the 100% admission ratio shows; and the packets of the remaining half of the legitimate ADs ("Public") obtained higher admission ratio along the public paths because the use of the private path reduced bandwidth contention along the public paths. This result consolidates how destination ADs are able to protect their valued customers' traffic from DDoS attacks in STRIDE.

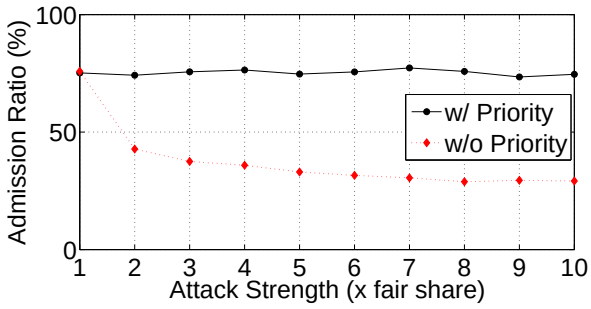


Figure 6: Effects of attack strength.

While STRIDE enables private parties to use private paths to avoid congested static paths, it also protects legitimate ADs' traffic from large-scale DDoS attacks via packet prioritization: i.e., capability requests made

[5]We construct the private path from the simulation topology

through the static up-paths would have a higher priority than others through the best-effort up-paths. To examine this, we use the following simulation scenario: the attack strength is increased (by adding more attack sources within contaminated ADs) up to 10 times the bandwidth of the static up-paths; source ADs put the high priority marking on their outbound packets such that the bandwidth of high priority packets would not exceed that of the static up-paths (e.g., if the attack strength grows 10 times, 90% of attack packets would have a low priority marking). Legitimate sources, on identifying congestion on static down-paths, use the best-effort down-paths; and attack sources use the same path selection strategy as that of the legitimate sources to maximize their attack influence. The above attack scenario is the strongest one for the given number of legitimate and attack sources since all packets that originate from 100 legitimate ADs and 300 contaminated ADs compete for the bandwidth of public paths.

Figure 6 shows that even if the attack sources grow their strength, their effects on legitimate traffic are marginal: attack sources, regardless of their strength, can consume bandwidth proportional to their fair share both on the static and the best-effort channels. Meanwhile, 25% of legitimate packets sent through the static down-paths reach their destination without loss, and the other legitimate packets (i.e., 75% of them) sent through the best-effort channel reach the destination with a ratio close to 66.7%. Overall, 75% of the legitimate requests overcome the massive DDoS attack whose total send rate is 30 times higher than that of the legitimate ones, even if no distinction between legitimate and attack packets can be made. The figure also shows that without packet prioritization, the admission ratio of legitimate packets decreases as attack strength grows. This result demonstrates the effectiveness of using static up-path and packet prioritization in STRIDE.

## 6.2 Flow Bandwidth Guarantees

STRIDE's bandwidth guarantees effectively isolate the bandwidth of attack traffic from that of legitimate traffic, if they follow different paths. We show this bandwidth isolation via large-scale simulations. For realistic simulations, we construct simulation topologies using a CAIDA SkitterMap [4], attach 10,000 legitimate sources (hosts) to 200 ADs proportional to the AD size, and attach attack sources (hosts) to 100 ADs. The actual number of ADs in a simulation topology can be slightly different, since we probabilistically sample paths from the SkitterMap for the given number of ADs. Legitimate sources control their packet sending rate based on the TCP congestion control mechanism, while attack flows send constant, high-rate traffic to flood a target link (i.e., a router) through which all destinations can be reached. We increase the attack size from 10K to 100K to compare STRIDE's bandwidth guarantees with those of a per-flow fair-sharing based mechanism. We use as the baseline the scenario with no defense mechanism.

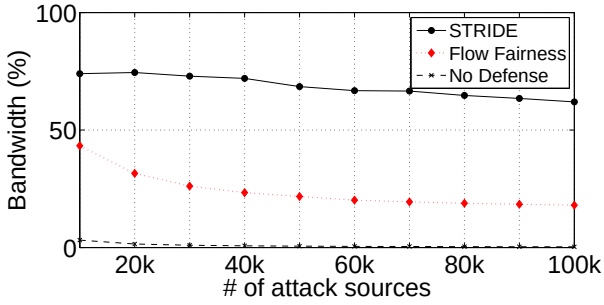


Figure 7: Bandwidth used by legitimate flows.

Figure 7 shows the bandwidth used by the legitimate flows that originate from clean ADs (i.e., ADs that do not contain attack sources). With no defense in the baseline scenario, the legitimate flows do not survive. This illustrates the lethality of DDoS attacks. When per-flow fair-sharing bandwidth control is employed at the target router, attack flows cannot completely exhaust the target's link bandwidth. However, the attack effects grow proportionally to the attack size, thereby reducing the bandwidth used by legitimate flows. We note that the bandwidth used by legitimate flows is substantially higher than their fair share because we implement the mechanism such that legitimate flows (that are responsive to a packet drop) are always prioritized than attack flows whenever they compete for the extra bandwidth unused by other flows. STRIDE provides consistent bandwidth guarantee to legitimate traffic under different attack sizes, which evidently proves the effectiveness of path bandwidth isolation. The bandwidth of legitimate flows decreases slightly as the attack size grows, because (1) the extra bandwidth that is not fully used by some paths [6] is shared by other flows and (2) as the number of contaminated ADs increases, the number of clean ADs decreases. Next, we increase the number of contaminated ADs by 10 up to 200 ADs. As one can imagine, the bandwidth to legitimate flows decreases as Figure 8 shows. However, we argue that the effects of attack dispersion across many ADs are marginal because the number of activated paths cannot increase indefinitely in STRIDE.

## 6.3 Throughput

STRIDE introduces additional computational work for capability (or opaque field) verification. To gauge the computational overhead of STRIDE, we measure

[6]Since we use TCP flow sources, allocated bandwidth may not be fully utilized by those sources.

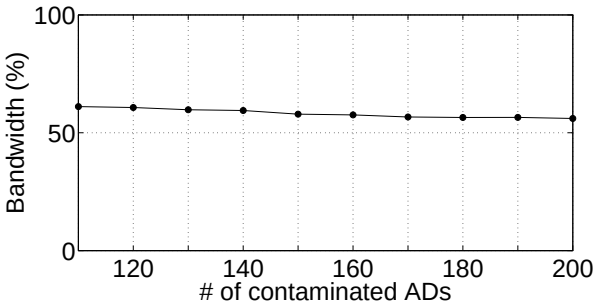


Figure 8: Bandwidth used by legitimate flows on increasing the number of contaminated ADs.

the throughput of a STRIDE router via a software prototype in Linux with various packet sizes and compare the result with that of the default IPv4 forwarding. We implement a STRIDE router as a user-space process using the Click Modular Router [16] running on Ubuntu 10.04 32-bit Desktop OS, which verifies the corresponding capability in each packet. The capability generation and verification is implemented using CBC-MAC with AES-ni. We implement the STRIDE router on off-the-shelf servers with an Intel Xeon E5640 CPU (four 2.66 GHz cores with 5.86 GT/s QuickPath Interconnect, 256KB L1 cache, 1MB L2 cache, 12MB L3 cache, and 25.6 GB/s memory bandwidth) and 12G DDR3 RAM. The servers have Broadcom NetXtreme II BCM5709 Gigabit Ethernet Interface Cards. We perform the measurement with a simple topology where a source and a destination are directly attached to a STRIDE router. We use the NetPerf benchmark [3] running on the source and the destination to stress-test the STRIDE router.

As described earlier, STRIDE forwards packets based on the interface identifier in the packet header, hence no additional overhead will be incurred in forwarding table lookup like in today's networks. Meanwhile, the IPv4 forwarding in our experiments would produce the best throughput that it can achieve since the Forwarding Information Base (FIB) has only one entry in our network configuration.

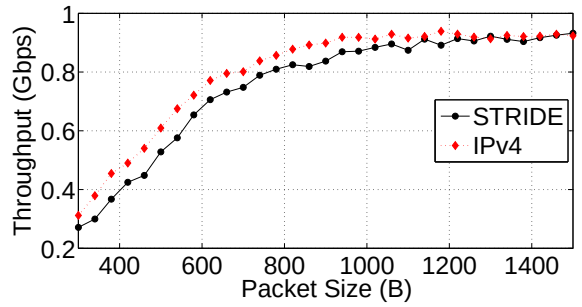


Figure 9: Throughput vs. packet size.

Figure 9 shows that for small packets, both IPv4 and STRIDE routers highly underutilize the link bandwidth while the IPv4 packet forwarding outperforms that of STRIDE; for large packets, they utilize more than 90% of the link capacity almost identically. In practice, the small packet overhead becomes negligible since they account for less than 10% of bandwidth yet most of remaining packets are full sized [5]. Hence, the STRIDE protocol, while providing strong security properties, would have far less packet forwarding overhead than the current IPv4 protocol.

# 7. DISCUSSION

## 7.1 Inter-TD Traffic Guarantees

While STRIDE provides the strongest guarantees for communication within a TD, many of the properties also translate for communication between TDs. Assuming no congestion on high-capacity links between TDs, static and dynamic channels obtain the same guarantees within each TD as channels that are only within a single TD. Only best-effort channels are getting lower guarantees, as there is no explicit indication that the receiver desires the communication. As a consequence, establishing a connection to a public service that is under attack will be challenging for an external host. However, as soon as the service receives one initial packet and desires serving that client, it can set up a dynamic channel with protected bandwidth guarantees.

## 7.2 Malicious ADs inside TD

ADs within a TD may get compromised and become malicious by dropping traffic or injecting malicious traffic. Although this is relatively unlikely in well-administered ADs, attackers can nevertheless exploit software vulnerabilities in routers or administrative workstations.

Defending against packet dropping can be accomplished through the multi-path nature of STRIDE, where a sender selects paths that avoid the AD after identifying that misbehaving AD (e.g., through some existing detection protocol [38]). Defending against malicious traffic injection is accomplished through neighborhood monitoring, where ADs enforce the bandwidth guarantees that they promised in their path construction beacons. If any AD sends more traffic than their allocated share, the AD must be malicious or misconfigured and the neighboring AD can block the offending traffic. In case an AD does not sufficiently monitor the amount of traffic it forwards from its customers, it may exceed its own allocations and risk getting caught by its neighbor.

In a more sophisticated attack, a malicious AD could drop traffic from one customer and instead inject traffic for another customer, thus potentially causing congestion elsewhere in the network. For this attack to work, however, the AD would need to forge packet headers with correct authenticators, otherwise, downstream ADs would simply drop the packets. Thus, the mali-

cious AD could copy existing packet headers with the goal of causing congestion elsewhere. Such forgery attacks are serious, and if other ADs detect such behavior, they can report it to the TD Core and have the AD revoked. Since all ADs of a TD are within a uniform legal environment, such mitigation mechanisms are effective. As a technical defense, once the malicious AD is detected, hosts can avoid the malicious AD by selecting paths that do not traverse that AD.

## 8. RELATED WORK

A plethora of network-layer DDoS defense mechanisms can be classified, in large, into two categories, namely, router-level bandwidth control and architectural extensions.

**Router-level bandwidth control.** Typical approaches to network-layer DDoS defense identify attack flows and either filter these flows or limit their bandwidth.

*Filtering.* Filtering approaches [7, 19] install filters against attack sources near their origins (i.e., source ADs) to prevent collateral damage of attack traffic. This would essentially require trust establishment between ADs and rely on source ADs's cooperation that would incur substantial overhead for managing flow state and packet inspection. In contrast, STRIDE naturally facilitates trust establishment between ADs thanks to the underlying SCION trust domains. Network-layer capability schemes [33,36] enable routers to perform stateless filtering without needing any trust on other routers, but are vulnerable to the DoC attack [8]. Though Portcullis [25] solves the DoC attack, it requires high computational overhead even on benign source hosts.

*Bandwidth throttling.* Many bandwidth control mechanisms (especially the fair queueing mechanisms) proposed to date can be used to prevent some (malicious) flows from exhausting the network bandwidth [20, 21, 24, 28, 32]. However, flow-level fair sharing does not provide any guarantees by design as the fair bandwidth becomes too low as more entities (e.g., flows) compete for a limited resource.

*Bandwidth guarantees.* Existing approaches [9, 10] aiming to provide bandwidth guarantees to flows fail in cases where all available bandwidth is exhausted. FLoc [18] differentiates legitimate flows from attack flows to provide differential bandwidth guarantees. Low-rate attack flows (e.g., Coremelt [29]), however, can often not be precisely distinguished from legitimate flows, thereby the lower bound of bandwidth may not be observed.

**Architectural support.** SCION inherently provides a default level of protection against DDoS attacks. For example, SCION's TD isolation prevents bots from attacking non-contaminated TDs; its periodic topology discovery and multipath by default enables agile path adjustment to avoid attacked areas. However, SCION itself does not provide any DDoS defense guarantee. Several other next-generation Internet architectures [23, 26, 34] have been proposed. Instead of providing intrinsic DDoS resilience, they aim to provide routing flexibility, path diversity [34], expressive routing policies [23, 26], etc. A line of multi-path routing protocols have also been proposed [13, 17, 22, 31, 35] to provide path diversity to the *source* nodes. Although the source nodes can utilize the path diversity to re-route around victim links/routers in a DDoS attack, the destinations are still left with little inbound traffic control. Furthermore, these protocols are still built on top of the current Internet, thus still suffering from the underlying weaknesses of today's Internet. For example, local identification of attack sources can be imprecise or impossible to counter large-scale botnet attacks, especially the Coremelt attack that does not target a specific service or endpoint.

## 9. CONCLUSION

A core goal of the STRIDE architecture is to achieve strong DDoS defense with relatively simple routers. In particular, we avoid per-flow state in the fastpath, asymmetric cryptographic operations, reliance on untrustworthy domains, and key establishment across ADs. Even with our relatively simple operations, we can achieve protection against DDoS from large botnets. Reflecting on the STRIDE architecture, we observe that measured trust in ADs that are located within the same legal environment providing viable prosecution helps to simplify the architecture and results in higher efficiency, meanwhile the untrustworthy ADs outside the trust domain cannot inflict damage against local within-trust-domain communication. We anticipate that STRIDE provides a useful point in the design space to study holistic network architectures with strong DDoS defense properties.

## 10. REFERENCES


[1] Arbor networks: Infrastructure security survey. `http://www.arbornetworks.com/sp_security_report.php`.

[2] Comcast data usage meter launches. `http://blog.comcast.com/2009/12/comcast-data-usage-meter-launches.html`.

[3] Netperf benchmark. `http://www.netperf.org/netperf/`.

[4] `http://www.caida.org/`.

[5] `http://www.caida.org/research/traffic-analysis/pkt_size_distribution/graphs.xml`.

[6] T. Anderson, T. Roscoe, and D. Wetherall. Preventing Internet denial-of-service with capabilities. In *Proceedings of ACM HotNets*, 2003.

[7] K. Argyraki and D. R. Cheriton. Active internet traffic filtering: real-time response to denial-of-service attacks. In *Proceedings of the USENIX Annual Technical Conference*, 2005.

[8] K. Argyraki and D. R. Cheriton. Network capabilities: The good, the bad and the ugly. In *ACM HotNets*, 2005.

[9] F. Bonomi and K. Fendick. The Rate-Based Flow Control Framework for the Available Bit Rate ATM Service. In *IEEE Network Magazine, vol. 9, no. 2*, 1995.

[10] R. Braden, L. Zhang, S. Berson, S. Herzog, and S. Jamin. Resource ReSerVation Protocol (RSVP) – Version 1 Functional Specification. RFC 2205 (Proposed Standard), Sept. 1997. Updated by RFCs 2750, 3936.

[11] C. Dixon, T. Anderson, and A. Krishnamurthy. Phalanx: Withstanding multimillion-node botnets. In *Proceedings of USENIX NSDI*, 2008.

[12] C. Estan and G. Varghese. New directions in traffic measurement and accounting: Focusing on the elephants, ignoring the mice. *ACM Trans. Comput. Syst.*, August 2003.

[13] P. B. Godfrey, I. Ganichev, S. Shenker, and I. Stoica. Pathlet routing. In *ACM SIGCOMM*, 2009.

[14] M. Handley and A. Greenhalgh. Steps towards a DoS-resistant internet architecture. In *Proceedings of the ACM SIGCOMM workshop on Future directions in network architecture (FDNA)*, Sept. 2004.

[15] A. Keromytis, V. Misra, and D. Rubenstein. SOS: An architecture for mitigating DDoS attacks. *IEEE Journal on Selected Areas in Communications (JSAC)*, 2004.

[16] E. Kohler, R. Morris, B. Chen, J. Jannotti, and M. F. Kaashoek. The Click modular router. *ACM Transactions on Computer Systems*, 2000.

[17] N. Kushman, S. Kandula, D. Katabi, and B. M. Maggs. R-BGP: Staying Connected In a Connected World. In *USENIX NSDI*, 2007.

[18] S. B. Lee and V. Gligor. FLoc: Dependable link access for legitimate traffic in flooding attacks. In *IEEE ICDCS*, 2010.

[19] X. Liu, X. Yang, and Y. Lu. To filter or to authorize: network-layer dos defense against multimillion-node botnets. In *SIGCOMM*, 2008.

[20] X. Liu, X. Yang, and Y. Xia. NetFence: Preventing internet denial of service from inside out. In *Proceedings of the ACM SIGCOMM*, 2010.

[21] R. Mahajan, S. M. Bellovin, S. Floyd, J. Ioannidis, V. Paxson, and S. Shenker. Controlling high bandwidth aggregates in the network. *Comput. Commun. Rev.*, 2002.

[22] M. Motiwala, M. Elmore, N. Feamster, and S. Vempala. Path splicing. In *ACM SIGCOMM*, 2008.

[23] J. Naous, M. Walfish, A. Nicolosi, D. Mazires, M. Miller, and A. Seehra. Verifying and enforcing network paths with ICING. In *ACM CoNEXT*, 2011.

[24] R. Pan, B. Prabhakar, and K. Psounis. Choke, a stateless active queue management scheme for approximating fair bandwidth allocation. In *INFOCOM*, 2000.

[25] B. Parno, D. Wendlandt, E. Shi, A. Perrig, B. Maggs, and Y.-C. Hu. Portcullis: Protecting connection setup from denial-of-capability attacks. In *Proceedings of the ACM SIGCOMM*, Aug. 2007.

[26] B. Raghavan and A. C. Snoeren. A system for authenticated policy-compliant routing. In *ACM SIGCOMM*, 2004.

[27] I. Stoica, D. Adkins, S. Zhuang, S. Shenker, and S. Surana. Internet indirection infrastructure. In *Proceedings of ACM SIGCOMM*, Aug. 2002.

[28] I. Stoica, S. Shenker, and H. Zhang. *Core*-stateless fair queueing: Achieving approximately fair bandwidth allocations in high speed networks. In *SIGCOMM*, 1998.

[29] A. Studer and A. Perrig. The coremelt attack. In *Proceedings of the 14th European Symposium on Research in Computer Security (ESORICS)*, Sept. 2009.

[30] M. Walfish, M. Vutukuru, H. Balakrishnan, D. Karger, and S. Shenker. DDoS defense by offense. In *Proceedings of the ACM SIGCOMM*, Sept. 2006.

[31] W. Xu and J. Rexford. MIRO: Multi-path Interdomain Routing. In *ACM SIGCOMM*, 2006.

[32] Y. Xu and R. Guérin. A double horizon defense design for robust regulation of malicious traffic. *SecureComm*, 2006.

[33] A. Yaar, A. Perrig, and D. Song. SIFF: A stateless internet flow filter to mitigate ddos flooding attacks. In *Proceedings of the IEEE Symposium on Security and Privacy*, May 2004.

[34] X. Yang, D. Clark, and A. W. Berger. NIRA: a new inter-domain routing architecture. *IEEE/ACM Trans. Netw.*, 2007.

[35] X. Yang and D. Wetherall. Source selectable path diversity via routing deflections. In *ACM SIGCOMM*, 2006.

[36] X. Yang, D. Wetherall, and T. Anderson. A DoS-limiting network architecture. In *Proceedings of the ACM SIGCOMM*, Aug. 2005.

[37] X. Zhang, H.-C. Hsiao, G. Hasker, H. Chan, A. Perrig, and D. G. Andersen. SCION: Scalability, control, and isolation on next-generation networks. In *Proceedings of the*

*IEEE Symposium on Security and Privacy (Oakland)*, May 2011.

[38] X. Zhang, Z. Zhou, H.-C. Hsiao, T. H.-J. Kim, A. Perrig, and P. Tague. ShortMAC: Efficient data-plane fault localization. In *Proceedings of the Networked and Distributed System Security Symposium (NDSS)*, 2012.

# APPENDIX

## A. SECURITY ANALYSIS

In this analysis, we first show how STRIDE achieves the desired goals as mentioned in Section 3.2. We then provide the estimation analysis on how STRIDE routers can divide their link capacities to three traffic classes and how much static bandwidth can be allocated to a path (Appendix A.2). We also describe how STRIDE mitigates known DDoS attacks that most previous work cannot mitigate (Appendix A.3).

### A.1 Guarantees, Robustness, and Path Control

#### *A.1.1 Goal 1: Precise Bandwidth Guarantees*

We discuss how STRIDE provides precise bandwidth guarantees for both public services and private end-to-end communication. Specifically, we show that STRIDE achieves (a) strong long-term static-class bandwidth guarantee, which ensures (b) strong end-to-end connection setup guarantee for private services and weaker guarantee for public services, and finally leads to (c) conditional strong end-to-end flow bandwidth guarantee. Note that the arguments for public services and private end-to-end communication differ only in (b). A *strong* guarantee is independent of variables such as number of bots, while a *weaker* guarantee has a linear dependency on some variable. A *conditional* guarantee relies on the accomplishment of another guarantee.

**(a) Strong long-term static-class bandwidth guarantee.**

In the current Internet, route advertisements flow from an edge customer AD up to the provider ADs and then down to other edge ADs. Such customer-initiated, arbitrary end-to-end routing advertisements prevent provider ADs from allocating bandwidth to end-to-end paths along the routing advertisements, because a customer may not know the total up-link capacity of its providers and the total number of customers its providers have.

Such This unpredictability, however, can be eliminated with STRIDE, thanks to the core-driven bandwidth allocation. As a result, STRIDE enables end-host ADs to establish bandwidth-guaranteed, long-term static half paths to the TDC.

**(b) End-to-end connection setup guarantee.**

To setup a bandwidth-guaranteed end-to-end connection, a source has to deliver its *first* packet (a capability request packet) to the destination. STRIDE provides a strong connection setup guarantee, or a first packet delivery guarantee, for both public services and private communication. For private communication, thanks to the support of private paths, the destination can selectively disclose some of its static down-paths to reserve static bandwidth for preferred sources. For example, a valued Amazon customer could get a 10 Kbps private down-path for sending capability requests to Amazon. This bandwidth policy can be enforced by the customer's host ISP.

However, securing the first packet delivery is challenging when the destination is publicly accessible. To address this challenge and mitigate attacks against first packets, STRIDE assigns levels of priority to capability requests to limit the attacker's power by forcing the attacker to wait longer or to exhaust its own up-path bandwidth, as described in Section 5 Step ⑤. In a nutshell, an AD prioritizes a capability request if the request traversed from a static up-path or an uncongested up-path. As a result, an attacker attempting to crowd out every legitimate capability request in a down-path is trapped into a dilemma: on one hand, the attacker needs to send an excessive amount of traffic using its own static paths; on the other hand, he has to keep his own static paths uncongested.

We formulate the waiting time before getting the first packet delivered. The result shows that the waiting time is linear to the number of contaminated ADs. For simplicity, assume each endpoint AD has $k$ up-paths and $k$ down-paths, where each half path supports $B_s$ units of static-class bandwidth and $B_e$ units of best-effort-class bandwidth. Assume that each capability request takes 1 unit bandwidth and each host can send at most $q$ requests. Each endpoint AD on the up-paths set the over-use bit in packets once the link capacity exceeds $B_s$. Let $n_i$ be the number bots in $\text{AD}_i$, $1 \leq i \leq M$, $M$ be the total number of ADs, and $\vec{n}$ be an array of the number of bots in each AD (i.e., $\vec{n} = \{n_1, n_2, \cdots, n_M\}$). Also, let $m$ be the number of contaminated ADs ($\text{AD}_i$ is contaminated iff $n_i \neq 0$). Now we model the waiting time $t(\vec{n})$, expressed as the number of setup trials, as a function of botnet size and distribution. The goal of the botnet is to generate excessive attack traffic, $B_{att}(\vec{n})$, to saturate the down-path bandwidth, $B_{down}$. Because of limited bandwidth per path, $B_{att}(\vec{n}) = \sum_{i=1}^{M} \min\{n_i \cdot q, B_s \cdot k\} \leq m \cdot B_s \cdot k$, and $B_{down} = (B_e + B_s) \cdot k$. Let $B_{legit}$ be the aggregate bandwidth of legitimate requests. When the sum of attack traffic ($B_{att}$) and legitimate traffic ($B_{legit}$) exceeds the down-path's capacity ($B_{down}$), a packet is randomly dropped with a probability $p(\vec{n}) = \frac{B_{down}}{B_{att}(\vec{n}) + B_{legit}}$. Hence, we can derive $t(\vec{n})$, which is a geometric distri-

bution with parameter $p(\vec{n})$:

$$t(\vec{n}) = \begin{cases} 1 & \text{if } B_{down} > B_{att}(\vec{n}) + B_{legit} \\ \frac{1}{p(\vec{n})} & \text{otherwise.} \end{cases} \quad (4)$$

Eq. 4 shows that the connection setup waiting time $t(\vec{n})$:

a) grows linearly to the number of contaminated ADs ($m$), and is bounded since $m \leq M$ and $M$ is a constant with respect to the number of bots. In contrast, in the current Internet, the waiting time is unbounded and can grow quadratically with $n$ (e.g., under a Coremelt attack [29]).

b) becomes lower if bots are concentrated in fewer ADs. To illustrate, consider two extreme cases where a) bots distribute uniformly over all AD domains, and b) all bots concentrated in one single AD. The attack traffic in case (1) is $B_1 = M \cdot \min\{n/M \cdot q, B_s \cdot k\}$ and in case (2) is $B_2 = \min\{n \cdot q, B_s \cdot k\}$. Since $B_1 \geq B_2$ for all $n$ and $M$, STRIDE further limits the attack strength under skewed botnet distribution.

c) decreases proportionally with the down-path bandwidth. This shows that STRIDE performs better regardless of the botnet distribution.

**(c) Strong end-to-end flow bandwidth guarantee.**

Given that the first capability request packet is secured (as described in (b)), the source can obtain a guaranteed share of end-to-end flow bandwidth, determined by the bottleneck link on the path. We show that STRIDE's achievable flow bandwidth guarantee only depends on the number of bots in the source and destination AD domains, thanks to STRIDE's fine-grained localization and isolation. In the analysis, we consider the worst case scenario where every host machine in the world is compromised except a source $S$ and a destination $D$, and the attacker clogs a bottleneck link by congesting an intermediate AD. The endpoint AD domains $\text{AD}_S$ and $\text{AD}_D$ contain $n_S$ and $n_D$ bots, respectively. Both $\text{AD}_S$ and $\text{AD}_D$ maintain $k$ half paths, each of which has $B$ units of dynamic-class bandwidth, and perform per-host fair sharing. Thus, after establishing a connection, $S$ and $D$ can obtain a flow bandwidth guarantee as $\min\{\frac{B \cdot k}{n_S+1}, \frac{B \cdot k}{n_D+1}\}$, given that $D$ only accepts $S$'s connection setup requests. In practice, $n_s$ and $n_d$ can be bounded or close to zero as the sender has strong incentives to subscribe to an non-contaminated AD. Therefore, STRIDE can provide guaranteed fair share dynamic allocation for legitimate requests.

#### A.1.2 Goal 2: Robustness and Efficiency

STRIDE avoids per-flow router state for data forwarding, because every packet carries forwarding information in opaque fields. To provide bandwidth guarantees, STRIDE requires only endpoint ADs to monitor per-flow traffic, which is feasible and has been implemented by major ISPs [2]. Additional checking can be done through probabilistic monitoring [12]. STRIDE requires no symmetric cryptographic operations or key establishment across ADs. Consequently, such a simple router design can prevent attacks that attempt to exhaust router resources.

#### A.1.3 Goal 3: Informed Path Control to the Edge

In the current Internet, ADs cannot efficiently avoid congestion as they have limited path diversity and control and lack any congestion information at remote ADs. In STRIDE, however, endpoint ADs can identify the congested area, thanks to 1) real-time announcement of available bandwidth in PCBs, 2) bottleneck pointer in the capability requests, and 3) congestion bits in dynamic capabilities. After identifying the congested area, STRIDE's flexible design enables endpoint ADs to make intelligent path selection to efficiently avoid congestion.

### A.2 Estimation Analysis

Using estimations, we show how STRIDE routers can divide their link capacity to the three traffic classes, and how much static bandwidth can be allocated to a path.

**Division of link capacity to three traffic classes.** We provide guidelines that an AD can follow to divide its total link capacity to three traffic classes: static, dynamic, and best-effort. First, given that the current real-world link utilization is mostly below 30% based on the CAIDA dataset [7], allocating 30% of the link capacity to the best-effort class would satisfy legacy Internet traffic in most cases. Subsequently, assuming the static and dynamic classes are allocated $s$ and $d$ fractions of the link capacity, respectively, the following conditions should hold:

$$s + d = 1 - 30\% \quad (5)$$

$$40\text{Gbps} \times s > 500\text{Kbps} \times 10000 \quad (6)$$

The first condition ensures a link will not be overloaded when each bandwidth class is being fully utilized. The second condition assumes an OC-768 link capacity (40Gbps) to be divided among around 10000 paths[8], and requires the static bandwidth allocated to each path be no less than 500 Kbps. Based on these guidelines, a reasonable example allocation is to divide 5 – 15%, 70 – 75%, and 10 – 20% link capacity to the static, dynamic, and best-effort traffic classes, respectively. In practice, an AD can adjust the numbers in the conditions based on its own link capacity, number of current paths that the AD supports, etc. Furthermore, when the static and dynamic bandwidth spectrum is not fully utilized, the

[7] CAIDA. `http://as-rank.caida.org/data/`

[8] We construct a US TD with around 2200 endpoint ADs based on the CAIDA dataset, and observe that if each provider AD can support 10000 paths, each endpoint AD can choose up to 10 paths.

best-effort traffic can always take up all the bandwidth that is currently available.

**Allocation of static bandwidth per path.** We address how much static bandwidth can be allocated to a half-path with a back-of-the-envelope analysis as follows. Assume that the Internet backbone uses OC-768 links, each with 40 Gbps bandwidth capacity. For illustration purposes, we consider OC-1 bandwidth as the minimum bandwidth unit for splitting. Then, an OC-768 would be split into 768 parts, each with roughly 50 Mbps bandwidth. If an AD further allocates 10% of the link capacity (or each bandwidth share) for the static bandwidth and a bandwidth share can support 10 activated paths, we derive that each activated path can obtain 500Kbps for its guaranteed static bandwidth. In other words, an endpoint AD that is subscribing OC-1 capacity from its provider will be able to activate ten 500Kbps paths.

### A.3 Resilience against Strong Attacks

**Denial-of-capability (DoC) attacks and link flooding.** In a DoC attack [7], bots flood the connection setup channel to prevent capability request packets from reaching the destination (i.e., the source's waiting time is unbounded). DoC attacks greatly compromise the effectiveness of most capability-based DDoS defense mechanisms, which assume a successful delivery of capability request packets. One approach to mitigate DoC attacks requires senders to demonstrate that they had to wait because they computed some puzzle [25], hence reducing the waiting time of legitimate senders from unbounded to linear. However, such puzzle-based schemes introduce high computational complexity to senders. In contrast, STRIDE utilizes packet marking and different levels of packet priority to efficiently achieve a linear waiting time guarantee under DoC attacks, as analyzed in Sec A.1.1.

**Coremelt attacks.** Conventionally, the DDoS victim is the destination of such unwanted attack traffic. However, in a Coremelt attack [29], $n$ bots send traffic between each other causing *quadratic* (i.e., $O(n^2)$) number of flows to congest a network link or router. Traditional capability-based DDoS systems rely on the receivers to identify unwanted flows. As a result, they cannot mitigate Coremelt attacks because all attack flows are "wanted" by the compromised destinations. In contrast, STRIDE effectively limits Coremelt attacks in the following three aspects:

*1. Limited number of paths.* In STRIDE, bots in $m$ ADs can cause at most $O(m \cdot k)$ traffic going through a targeted victim AD, where $k$ is a bounded value representing the number of static paths used by an AD. The reason is that each AD can at most access $k$ activated paths, and each of these paths has an allocated bandwidth that is constant with respect to the number of bots in the AD.

*2. Flow bandwidth guarantee.* The $k$-path policy constrains the amount of attack traffic perceived by a victim AD. Another significant question is to ask, from the end-hosts' point of view, that how much flow bandwidth a pair of legitimate hosts can obtain when one or more intermediate ADs are under Coremelt attacks. As analyzed in Sec A.1.1, STRIDE's end-to-end bandwidth flow guarantee ensures that a destination receives a fair share of bandwidth, reducing the attack strength from quadratic (to the number of bots in total) to linear (to the number of bots in the endpoints' ADs).

*3. Re-routing.* Based on congestion bits and bottleneck pointers, legitimate ADs or hosts can re-route their packets to avoid targeted area, further mitigating Coremelt attacks and identifying potential bots. Flows that still travel through the targeted area are likely to be malicious.

## B. DISCUSSION ABOUT SHORTCUTS AND ASYMMETRIC PATHS

### B.1 Path Shortcuts over Peering Links

To achieve short end-to-end paths, SCION proposes the notion of path shortcuts over peering links, which are identified through additional peering links of an AD that are announced with path construction beacons [37]. If both the up-path and down-path announce the same peering link, hosts can use that shortcut to avoid forwarding through the TD core even though the two paths do not intersect in any AD.

All the mechanisms proposed in this paper also apply to peering links. During PCB forwarding, ADs also announce available static and dynamic bandwidth for peering links. For activating static paths, the bandwidth is only reserved on the actual path, however, the peering link can be used if the static bandwidth is available. For guaranteed bandwidth, however, a detour through the TD core needs to be taken. In the case of dynamic channels, the bandwidth allocated and the fair-share resource allocation over peering links is analogous to the regular paths described in the paper.

### B.2 Asymmetric Paths

In practice, network links may be directional or asymmetric with different bandwidths in the two directions. STRIDE can flexibly accommodate asymmetric paths with minimal modifications as follows. To request a dynamic capability along an asymmetric path, the sender host $S$ puts in the request header both the forward (from $S$ to $D$) and backward paths (from $D$ to $S$). Dynamic path capabilities can be requested on both the forward and return paths, and sent back to the other party through sufficient space allocated in the packet header.

Not only the path, but also the bandwidth requests may be asymmetric, as in downloads the client-to-server bandwidth is 1 to 2 orders of magnitude smaller (acknowledgment packets are much smaller than data packets). In this case, STRIDE supports asymmetric bandwidth allocations, where a unidirectional capability is requested with different amounts depending on the direction.

## C. BANDWIDTH OVERBOOKING

In Section 5.1 ①, we introduce bandwidth overbooking for simultaneous enhancement of path quality and diversity, but with possible denial of path activation. To mitigate such an issue, we suggest an appropriate overbooking ratio by analyzing the relationship between an overbooking ratio and the corresponding probability of path activation denial as follows:

For AD $AD_p$, let $I_i$ and $E_j$ represent an $i^{th}$ ingress interface from the providers ($0 \leq i \leq l$) and a $j^{th}$ egress interface to the customers ($0 \leq j \leq m$), respectively. respectively. Let each ingress interface connect to all $m$ egress interfaces. We assume that $m$ interfaces connect to $n$ customer ADs (i.e., each customer AD has $\frac{m}{n}$ links to $AD_p$). Then, each customer AD has $\frac{l \cdot m}{n}$ to the TDC through $AD_p$. In this setting, if each customer AD randomly selects $k$ paths to the TDC, the probability that customer ADs select $I_i$ more than $t$ times would be: $P_{I_i}(t) \approx 1 - \sum_{i=0}^{t} e^{-\lambda} \cdot \frac{\lambda^i}{i!}$, where $\lambda = \frac{n \cdot k}{m}$. This implies that $I_i$'s bandwidth needs to be allocated to $t$ egress interfaces (out of $m$ interfaces), which would increase per-path bandwidth allocation by $\frac{t-\beta}{\beta}$, where $\beta = \frac{n \cdot k}{l}$ is the average number of activated paths through $I_i$. If $t \gg \frac{n \cdot k}{l}$, sufficient path diversity (as much as $\frac{t \cdot l}{n \cdot k}$) is provided to customer ADs.

As a result, $AD_p$ may determine $t$ such that the probability of the denial of path activation does not exceed some threshold $P_{th}$ (i.e., $P_{I_i}(t) \leq P_{th}$). For example, $P_{th} = 0.2$ means that 80% of path activation requests would be accepted on average; hence, the expected number of trials for successful path activation becomes 1.25. That is, $P_{th}$ determines the number of requests that should be made by an endpoint AD until successful path activation.

## D. PATH SELECTION

We investigate the availability of bandwidth-guaranteed paths at endpoint ADs by analyzing the current Internet connectivity and provide an evidence showing that using a probabilistic path selection strategy, an endpoint AD can activate a *high-quality* bandwidth-guaranteed path *efficiently*.

In STRIDE, ADs overbook bandwidth during PCB propagation in order to make higher bandwidth allocation while providing sufficient path diversity. This necessarily introduces contention among endpoint ADs in path activation. We evaluate the endpoint ADs' path selection with respect to two metrics below.

- activation overhead: the average number of trials[9] for successful path activation.
- path quality: the sum of *advertised* bandwidth guarantees (as opposed to the activated guarantees by an endpoint AD) of the $k$ selected paths by an endpoint AD.[10]

We first investigate how endpoint AD's path selection policy affects successful path activation. We compare the activation overhead of three different path selection strategies: (1) highest-bandwidth-first, (2) random, and (3) probabilistic strategies. In the probabilistic strategy, an AD selects a path with a probability proportional to the bandwidth guarantee of a path.

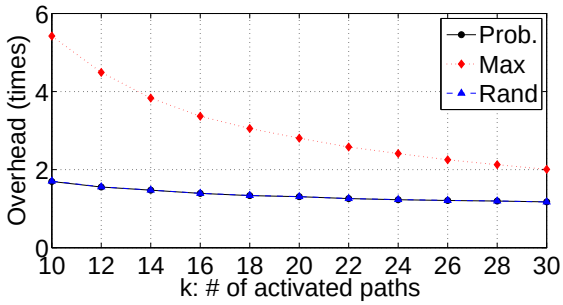


Figure 10: Activation overhead. Prob and Rand have almost identical overhead, hence their plots overlap.

Figure 10 shows the average number of trials until each endpoint AD successfully activates $k$ paths under the 10% overbooking ratio. With the probabilistic scheme ("Prob"), which is largely overlapped with the random scheme("Rand"), an endpoint AD activates a path in fewer than two trials even in the tight bandwidth allocation scenario (where the total bandwidth to the TDC is used up eventually) and needs fewer trials for a larger $k$ since intermediate ADs can grant more path activations that require low bandwidth guarantees. If endpoint ADs try to activate the highest-bandwidth path first, they would have much higher contention near the TDC, which delays their path activation ("Max").

ADs would naturally prefer a path that contains a higher bandwidth guarantee. Hence, we simulate a scenario where the endpoint AD maximizes the path quality by selecting the highest-bandwidth path first, and study the path qualities yielded by the probabilistic

[9]If an endpoint AD selects a path to activate but fails (the path's bandwidth is already fully subscribed), that AD will select another path to activate, which is named a trial.

[10]An endpoint AD can activate a path with a fraction of advertised guarantee to increase its path diversity to the TDC.

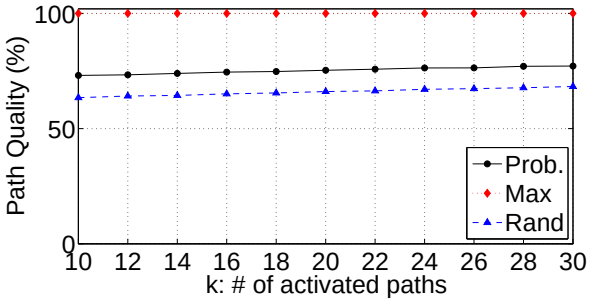


Figure 11: Path Quality.

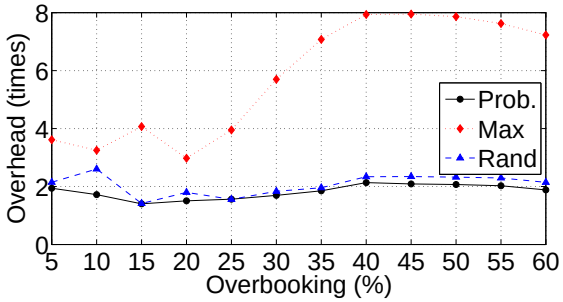


Figure 12: Overbooking.

and random selection schemes relative to the highest-bandwidth-first approach. Figure 11 shows the path quality of probabilistically and randomly selected paths relative to that of the highest-bandwidth paths. The probabilistically selected paths yield about 75% of the path quality in the highest-bandwidth-first approach, and have 10% higher quality than the randomly selected paths. While an endpoint AD has the highest path quality by selecting the highest-bandwidth path first, the AD is more likely to experience frequent contention with other ADs since higher quality paths would be fully subscribed earlier. Hence, probabilistic path selection would be a better choice both for successful path activation and for a better path quality.

Next, we change the overbooking ratio from 5 to 60% and observe how the ratio affects the number of activation trials. Figure 12 shows that the overhead of the probabilistic scheme decreases until 15% overbooking ratio and increases. With low overbooking ratio, endpoint ADs are provided more paths while those paths do not highly overlap. However, as ADs overbook their bandwidth to a further extent, more paths overlap, which necessarily causes activation denial. The probabilistic scheme shows the least overhead for the entire overbooking range tested and the random scheme shows similar results. Meanwhile the highest-bandwidth-first scheme results the highest overhead since more contentions are concentrated on the highest quality paths. The results strongly encourage endpoint ADs to employ the probabilistic path selection scheme to mitigate activation overhead in the face of ADs' bandwidth overbooking.