

ABSTRACT

Title of dissertation: Localizing the Effects of Link Flooding Attacks
in the Internet

Soo Bum Lee
Doctor of Philosophy, 2009

Dissertation directed by: Professor Virgil D. Gligor
Department of Electrical and
Computer Engineering

Malware-contaminated hosts organized as a “bot network” can target and flood network links (e.g., routers). Yet, none of the countermeasures to link flooding proposed to date have provided *dependable* link access (i.e., link access guarantees) for legitimate traffic during such attacks. Network-layer capabilities offer strong protection against link flooding by authorizing individual flows with unforgeable credentials (i.e., capabilities). However, network-layer capabilities are insufficient for dependable link access, for several reasons: (1) the capability-setup channel is vulnerable to flooding attacks that prevent legitimate clients from acquiring capabilities; i.e., Denial of Capability (DoC) attacks, (2) compromised attack sources that have acquired capabilities in a legitimate way can flood the privileged channel reserved for capability carrying packets, and (3) the global effects of flooding attacks are still unavoidable with “per-flow” based capabilities.

In this dissertation, we present a router-level design that confines the effects of link flooding attacks to specified locales or neighborhoods (e.g., one or more

administrative domains of the Internet) based on network-layer capabilities. Our design provides differential guarantees for access to network links that favor packets from uncontaminated domains by attack sources (e.g., bots) and yet do not deny access to packets from contaminated domains. For connection-request packets (i.e., capability requests), differential access guarantees are defined as the probabilistic lower bounds for link access: requests from uncontaminated domains have higher probabilistic lower bounds for link access than those from contaminated domains. For all other packets, differential access guarantees are defined in terms of the bandwidth allocated to packet flows; i.e., flows of malware-uncontaminated domains receive higher bandwidth guarantees than flows of contaminated ones, and legitimate flows of contaminated domains are guaranteed substantially higher bandwidth than attack flows. Potential side-effects of attack flows (e.g., multiple congested links) are mitigated by a differential routing scheme, whereby flows of malware-uncontaminated domains are routed through less congested paths while those of contaminated domains are routed through the “pinned” default paths.

We present analytical models for the proposed notions of dependable link access, and evaluate our router design both by comprehensive simulations under different attack scenarios and by comparisons with other flooding-defense schemes.

Localizing the Effects of Link Flooding Attacks in the Internet

by

Soo Bum Lee

Dissertation submitted to the Faculty of the Graduate School of the
University of Maryland, College Park in partial fulfillment
of the requirements for the degree of
Doctor of Philosophy
2009

Advisory Committee:
Professor Virgil D. Gligor, Chair/Advisor
Professor Charles B. Silio
Professor John S. Baras
Professor Gang Qu
Professor Mark A. Austin

© Copyright by
Soo Bum Lee
2009

DEDICATION

to my loving Hyun Jung,

Ji Hyun, Sang Hyuk,

and parents.

ACKNOWLEDGMENTS

First of all, I would like to thank my advisor, Professor Virgil D. Gligor for guiding me throughout my Ph.D course and providing invaluable insight on the various topics of network security. I feel highly honored to have been working with him during my Ph.D. program. It has been a great pleasure to perform my research under his supervision.

I would like to thank Professor Charles Silio and John Baras for being served as the committee members in both the proposal examination and the dissertation defense. I would like to thank Professor Gang Qu and Mark Austin for agreeing to be the committee members in the dissertation defense.

I wish to thank Professor John Baras for hosting CSHCN (currently Hynet) colloquium series for several years, which strongly motivated me to start and continue research in networking and security areas, and gave me chances to meet and discuss with distinguished people in those areas. I am indebted to KGSYS (Korean Graduate Student System Research Group) members, Dr. Joonhyuk Yoo, Dr. Minho Shin, Dr. Seungjoon Lee, Jihwang Yeo and Minkyoung Cho, for giving me invaluable advice not only on my research but on my life.

I have had the pleasure to work with Professor Gligor's group members: Dr. Gelareh Taban, Dr. Rakesh Bobba, and Dr. Ji Sun Shin. And, many colleagues in ECE and CS department have enriched my graduate study and life in many ways. I

have been able to spend a wonderful time during my Ph.D study thanks to those colleagues: Dr. Younggu Kim, Dr. Seung-Jong, Dr. Kejong Kim, Dr. Woocheol Jeon, Dr. Donghun Park, Dr. Jusub Kim, Dr. Seokjin Kim, Sangchul Song, Jookyung Lee, Hojin Kee, Inseok Choi, Kwangsik Choi, Keunmin Ryu, Kyungjin Yoo, Jeongho Jeon, Jaehwan Lee, Sungwoo Park, and all other students. I also thank my friend Dr. Jungsub Kim who has been around me for more than 15 years, and has helped and encouraged me to get through this long journey.

Most of all, I am deeply indebted to my parents for their countless love and support throughout the course of my life. Also, I am indebted to parents-in-law, sisters, and brothers for their love, understanding, support and encouragement. Words cannot express my gratitude to my loving Hyun Jung for her unwavering support, love, patience, encouragement and sacrifice, and to my lovely kids, Ji Hyun and Sang Hyuk, for the happiness they brought into my life.

Thank God.

Soo Bum Lee, 2009

TABLE OF CONTENTS

| | |
|--|------|
| List of Tables | viii |
| List of Figures | ix |
| List of Abbreviations | xii |
| 1 Introduction | 1 |
| 1.1 Dependable Connection Setup for Network Capabilities | 4 |
| 1.2 Dependable Link Access for Legitimate Traffic | 7 |
| 1.3 Collaborative Defense for Mitigating Attack Effects | 9 |
| 1.4 Organization | 13 |
| 2 Dependable Connection Setup for Network Capabilities | 14 |
| 2.1 Outline of the Chapter | 14 |
| 2.2 Background and Related Work | 14 |
| 2.3 Design Overview | 16 |
| 2.3.1 Threats | 17 |
| 2.3.2 Path Identification | 17 |
| 2.3.3 Link Access Guarantees | 19 |
| 2.3.3.1 Fair Queueing Revisited | 19 |
| 2.3.3.2 Path Aggregation | 21 |
| 2.4 Path Identification | 22 |
| 2.4.1 Unspoofable path-identifier | 23 |
| 2.4.1.1 Authenticator Distribution | 24 |
| 2.4.1.2 Origin Authentication | 25 |
| 2.4.2 Preventing Replay Attacks | 25 |
| 2.5 Dynamic Virtual Queueing | 27 |
| 2.5.1 Implementing Buffer-slot Preemption | 28 |
| 2.5.2 Probabilistic Guarantees | 30 |
| 2.5.3 Resource Requirements | 31 |
| 2.5.3.1 Request Packet Buffer | 31 |
| 2.5.3.2 Path-Identifier Accounting | 31 |
| 2.6 Path Aggregation | 32 |
| 2.6.1 Goodput Estimation | 32 |
| 2.6.2 Aggregation Problem | 35 |
| 2.6.3 Aggregation Algorithm | 37 |
| 2.7 Simulation Results | 39 |
| 2.7.1 Link-Access Guarantees | 41 |
| 2.7.2 Aggregation Effect | 43 |
| 2.7.3 Rolling Attacks | 45 |
| 2.8 Proof of Claims | 46 |
| 2.8.1 Proof of Guarantees | 46 |
| 2.8.2 Proof of Error Bound | 47 |

| | | |
|---------|--|-----|
| 3 | FLoc : Dependable Link Access for Legitimate Traffic in Flooding Attacks | 50 |
| 3.1 | Outline of the Chapter | 50 |
| 3.2 | Related Work | 51 |
| 3.3 | Design Overview | 53 |
| 3.3.1 | Assumptions | 53 |
| 3.3.2 | Bandwidth Guarantees in Normal Mode | 53 |
| 3.3.3 | Bandwidth Guarantees in Attack Mode | 55 |
| 3.4 | Modeling Dependable Link Access | 57 |
| 3.4.1 | Token-Bucket Model Revisited | 57 |
| 3.4.2 | Attack-Flow Identification and Confinement | 62 |
| 3.4.2.1 | Attack (Domains) Paths | 62 |
| 3.4.2.2 | Attack Flows | 63 |
| 3.4.2.3 | Confinement of Covert Attacks | 65 |
| 3.4.3 | Differential Bandwidth Guarantees | 66 |
| 3.4.3.1 | Attack-Path Aggregation | 67 |
| 3.4.3.2 | Legitimate-Path Aggregation | 68 |
| 3.5 | Router Design | 70 |
| 3.5.1 | Token-Bucket Activation and Router Queue Management | 70 |
| 3.6 | Simulation Results | 74 |
| 3.6.1 | Attack Confinement | 75 |
| 3.6.2 | Robustness of Bandwidth Guarantees | 76 |
| 3.6.3 | Effects of Different Path Delays on Bandwidth Guarantees | 77 |
| 3.6.4 | Differential Bandwidth Guarantees | 78 |
| 3.6.5 | Covert Attacks | 81 |
| 4 | A Collaborative Defense against Link Flooding Attacks | 87 |
| 4.1 | Outline of the Chapter | 87 |
| 4.2 | Related Work | 87 |
| 4.3 | Design Overview | 89 |
| 4.3.1 | Problem Statement | 89 |
| 4.3.2 | Assumptions | 91 |
| 4.3.3 | Multipath Routing | 91 |
| 4.3.4 | Path Pinning | 94 |
| 4.3.5 | Rate Control | 95 |
| 4.4 | Architecture | 96 |
| 4.4.1 | Route Controller | 97 |
| 4.4.2 | Route Management | 98 |
| 4.4.2.1 | Domain Discrimination | 99 |
| 4.4.2.2 | Rerouting | 100 |
| 4.4.2.3 | Path Pinning | 103 |
| 4.4.3 | Rate Throttling | 104 |
| 4.4.3.1 | Rate Allocation | 104 |
| 4.4.3.2 | Packet Marking | 106 |
| 4.4.3.3 | Rate Control | 106 |
| 4.4.4 | Control Message Format | 109 |

| | | |
|-------|---|-----|
| 4.5 | Simulation | 111 |
| 4.5.1 | Multipath Routing | 113 |
| 4.5.2 | Effects of Attack-Path Change | 116 |
| 5 | Conclusion and Future Work | 119 |
| 5.1 | Conclusion | 119 |
| 5.2 | Future Work | 121 |
| | Bibliography | 123 |

LIST OF TABLES

3.1 Flow bandwidth for different RTT_i s. \mathcal{F}_{LP}^g denotes the set of legitimate flows in Legitimate Paths and \mathcal{F}_{AP}^g denotes the set of legitimate flows in Attack Paths. ΔDelay is the attack paths' link delays relative to those of the legitimate paths. The mean and standard deviation of bandwidth are given in Mbps. 78

LIST OF FIGURES

| | | |
|-----|---|----|
| 2.1 | Path Identifier. R'_4 is the egress router of AS4 and R_3, R_2, R_1 are the ingress routers of AS3, AS2, AS1 respectively. R_C stands for the congested router. R'_4 writes the path-identifier to the packet heading to server S in AS1, and ingress routers on the path validate the marking. C_j is the capability issued by R_j . Each ingress router can validate the shaded part of the marking. | 19 |
| 2.2 | Path-identifier Authentication. ① Path-identifier written at the packet's origin (AS4) can be validated at any domain (e.g., AS2, AS1) in the presence of a non-marking domain(s) (e.g., AS3) on the packet's forwarding path. ② If the origin AS (e.g., AS4) does not participating in the marking, the first participant on the path (e.g., AS2) writes its marking and adds the incoming AS number (AS3) to distinguish the packets it forwards from the ones originating from it. ③ An invalid ASN-OAN pair (e.g., $OAN^\#$ represents a forged OAN at AS4) can be detected and filtered at any participating domain. | 26 |
| 2.3 | Counting Bloom Filter : In CBF, m counter arrays (a_1, \dots, a_m) are associated with m hash functions ($H_1(\cdot), \dots, H_m(\cdot)$). The counter value of S_1 in a_i is referenced by $a_i[H_i(S_1)]$ | 28 |
| 2.4 | Aggregation example. The numbers in the leaf and intermediate nodes represent the conformance estimates and the aggregation costs respectively. And, the parenthesized numbers in the table represent the reduced number of path-identifiers by aggregation. | 39 |
| 2.5 | Topology used in simulation. Legend: "d" is the number of sibling nodes (degree) and "h" is the tree height. | 39 |
| 2.6 | Request drop ratio of legitimate paths. Error bars represent 95% confidence intervals. | 40 |
| 2.7 | Request service probability of legitimate paths with respect to bandwidth utilization (ρ). The solid horizontal lines inside bars represent the probabilistic guarantees ($\mathcal{G}(\mathcal{S} , k, S_i)$). | 40 |
| 2.8 | Request drop ratio of legitimate paths with respect to attack location in the unbalanced tree. TVA(k) represents the result of TVA with queue-length k | 44 |
| 2.9 | Aggregation by protocol conformance: The request service ratio of legitimate paths increases as the fraction of bots becomes higher. . . | 44 |

| | | |
|------|---|----|
| 2.10 | Time variation of goodput ratio at the congested link. Legend: Error bars represent the minimum and maximum of goodput ratio. | 45 |
| 3.1 | TCP window variations and their aggregate effect at a congested router for unsynchronized, partially synchronized, and synchronized flows. Legend: Upper graphs: gray areas denote individual packet-flow rates. Lower graphs: grey areas illustrate three different distributions of the same aggregate-flow rate at a congested router. | 57 |
| 3.2 | Simulation Topology | 74 |
| 3.3 | Localization effects for three different attacks. Legitimate and attack paths are randomly chosen from 27 paths. Legend: For each path identifier, “Service” denotes the bandwidth received, and “Arrival” the bandwidth requested | 83 |
| 3.4 | Cumulative Distribution Function (CDF) of legitimate path flows’ bandwidth for various CBR attack rates. Note: For FLoc, the bandwidth of legitimate-path flows has nearly identical distribution independent of attack strength. In contrast, for Pushback and RED-PD, this bandwidth decreases significantly (i.e., the CDF curve moves left) as the attack strength increases. | 84 |
| 3.5 | Differential Guarantees: Legitimate vs. Attack Flows. Legend: The CDFs labeled by “Aggr” illustrate the results of path-identifier aggregation. LP and AP denote Legitimate and Attack Paths, respectively. | 85 |
| 3.6 | Differential bandwidth used at flooded link. (Flow rates of each attack source are shown on the horizontal axis.) | 85 |
| 3.7 | Bandwidth used at the flooded link under covert attacks. | 86 |
| 4.1 | Multipath Routing : AS3 is an uncontaminated domain whose traffic is to be directed to an alternative path, and AS4 is a contaminated domain that sends attack traffic. (a) AS3 selects a different provider (AS5) in forwarding packets to the congested link, instead of AS2 (which has shorter AS path length to AS1). (b) The provider AS (AS2) forwards the packets received from AS3 to a different path (i.e., through AS5). | 92 |

| | | |
|-----|---|-----|
| 4.2 | Route Control Messages: The congested router sends congestion notification (CN) message to its route controller (RC:B), and the message is forwarded to the route controller (RC:A) that can handle the requested message. The multipath routing (MP) message establishes a tunnel between ingress and egress routers; the path-pinning message (PP) suppresses route update message from the downstream domains; and the rate-throttling message (RT) makes responsible routers mark/drop packets. | 97 |
| 4.3 | Rate Limiting at a Congested Router. | 108 |
| 4.4 | Control Message Format. | 109 |
| 4.5 | Simulation Topology | 112 |
| 4.6 | Bandwidth used by individual source domains at the congested link when the send rate of the attack domain (i.e., S_1) is 200 Mbps. PBW stands for global deployment of path-bandwidth control at routers. | 114 |
| 4.7 | Bandwidth used by individual source domains at the congested link when the send rate of the attack domain is 300 Mbps. | 116 |
| 4.8 | Bandwidth used by source domains at the congested link. Legend: SP: Single-path Routing, MP: Multi-path Routing, SPP: SP with global per-path bandwidth control, MPP: MP with global per-path bandwidth control. The numbers following dashes represent the send rate of each attack domain. | 117 |
| 4.9 | Comparison of bandwidths used by individual source domains at the congested link for different attack send rates (i.e., 400 Mbps and 500 Mbps) and different routing policies (i.e., single-path and multi-path routing). | 118 |

List of Abbreviations

| | |
|-------|---|
| DoS | Denial of Service |
| DDoS | Distributed Denial of Service |
| AS | Autonomous System |
| BGP | Border Gateway Protocol |
| IGP | Interior Gateway Protocol |
| OSPF | Open Shortest Path First |
| IS-IS | Intermediate System–Intermediate Systems |
| MPLS | Multi Protocol Label Switching |
| RSVP | Resource Reservation Protocol |
| NAT | Network Address Translation |
| ISP | Internet Service Provider |
| ICANN | Internet Corporation for Assigned Names and Numbers |
| OAN | Origin Authentication Number |
| ASN | Autonomous System Number |
| CBF | Counting Bloom Filter |
| PIR | Path Identifier Record |
| PDHT | Packet Drop History Table |
| MTD | Mean Time to Drop |
| CBR | Constant Bit Rate |
| RC | Route Controller |
| CN | Congestion Notification |
| MP | Multi-Path routing |
| PP | Path Pinning |
| RT | Rate Throttling |

Chapter 1

Introduction

It is generally understood that DDoS attacks that flood Internet services require end-to-end solutions; i.e., absence of flooding in the underlying network links is insufficient to guarantee service access [Gli05]. It is equally well-understood that denial of service is *not* an end-to-end solvable problem; e.g, DDoS attacks that flood *public services* cannot be countered without handling flooding attacks at the network layer. Yet, most proposed solutions that attempt to provide service-access guarantees despite flooding attacks assume the existence of flood-free network links – a non-trivial assumption that has not been satisfied in the Internet to date. Traditional techniques for handling such flooding attacks at the network layer, including IP tracebacks [SP01,SWKA00,Bel00,YPS03,Goo02], IP ingress/egress filtering [Fer00], and legal/administrative remedies, though useful, are insufficient. For example, service-access requests can originate from zombies or bots that issue protocol-conforming requests from legitimate IP addresses and remote Internet locations outside jurisdictions that can deter flooding attacks. Hence, these attacks can evade most traditional techniques.

Handling flooding attacks against network links requires different solutions

from those at the application layer for at least two reasons. First, the typical mechanisms used at the application layer (e.g., mechanisms based on efficiently-verifiable request authenticity, on proof of work [ANL00, WR03, WVB⁺06], on proof of valid request scheduling [Gli05], on proof of human presence [vABHL03]) are impractical at this layer. Second, application layer mechanisms provide only *very weak* guarantees of access during flooding attacks as they are (at best linearly) dependent on the number global of attack sources [ANL00, Gli05, WR03, WVB⁺06]. This undesirable dependency cannot be removed by any of the network-layer defenses proposed to date. For example, recent “capability” based solutions, whereby distinct packet flows are separately authorized by capabilities obtained before flows are initiated [ARW03, YPS04, YWA05], cannot offer access guarantees because large-scale attacks (e.g., 1 – 2 million attack sources) can flood most chosen links with packets containing requests to obtain capabilities. And, in the absence of strong user/client authentication, zombies or bots can acquire capabilities for a target link and flood the link in a legitimate manner. Naturally, if link-access guarantees that are independent on the number of global attack sources cannot be obtained, then strong guarantees of access via network-layer capabilities become impossible.

Solutions that remove the strong flooding-freedom dependency on critical network links to a service (e.g., by providing many network links to the service and distributing flows randomly among them), such as SOS [KMR02] and its descendants [SCM⁺05, SK05, And03], require network overlays for specific web servers with distributed access points that authenticate client requests. Though useful, these solutions do not aim at providing access guarantees for individual network links. Such

guarantees would still be necessary for Internet infrastructure and other public services, for instance, which cannot authenticate remote requests (e.g., authentication servers themselves, domain name servers).

The central problem we address in this dissertation is that of “localization” of flooding attacks against network links. That is, we seek to provide guarantees of network-link access that depend only on attack sources in *defined locales* or neighborhoods (e.g., a local administrative domain or a set of domains in the Internet) and not on *all* possible attack sources. As a consequence, competing flows to flooded links that originate outside a contaminated locale should be unaffected, or only minimally affected, by a flooding attack. Hence, they should get better guarantees of network-link access than flows of contaminated domains. In effect, we seek to provide *differential guarantees* for network-link access that favor flows from administrative domains that are uncontaminated with attack sources instead of uniform but very weak guarantees.

In this dissertation, we present a router-level scheme that provide (1) connection setup guarantees for network capabilities, (2) bandwidth guarantees for legitimate flows carrying capabilities, and (3) route diversity for legitimate flows.

1.1 Dependable Connection Setup for Network Capabilities

Recent service-flooding attacks used a large number of compromised machines organized as a “bot” networks. Typical defense mechanisms that attempt to provide service-access guarantees despite such attacks assume absence of flooding in the underlying network links. Yet, a large scale attack (e.g., a “botnet” with millions of “bots”) can flood any chosen link in the Internet. In particular, defense mechanisms deployed at links near or at a network edge (e.g., Firewalls, IDSs) can be easily overwhelmed by such attacks. Worse yet, “legitimate-looking” attack packets can evade most of traditional techniques for handling address spoofing attacks at the network layer (e.g., IP tracebacks, ingress filtering).

“Capability” based solutions, whereby distinct packet flows are separately authorized by capabilities obtained before flow initiation [ARW03, YPS04, YWA05, YWA08], provide congested routers with an effective way to prioritize legitimate flows and filter out unwanted traffic. Though promising, these solutions are still vulnerable to flooding attacks targeting the *capability-setup channel*, known as the Denial of Capability (DoC) attacks [AC05b]. These attacks are possible because the initial capability-request packets are treated as “best-delivery-effort” packets, as opposed to the subsequent high-priority packets that carry capabilities. If DoC attacks cannot be countered, flow authorization via network-layer capabilities becomes impossible, and all access guarantees become meaningless at congested routers.

Previous solutions that attempt to protect capability requests from flooding

attacks (e.g., mechanisms based on aggregate request rates [YWA08] or on proof of work [PWS⁺07]), though useful, are insufficient to provide dependable link-access guarantees for legitimate capability requests. For example, a fair-queueing mechanism, which allocates buffer space to flow aggregates fairly based on a router’s confidence on precise identification of traffic origin [YWA05, YWA08], fails to provide *any* guarantee of link-access (viz., Section 2.7.1). Mechanisms based on proof of work (e.g., Portcullis [PWS⁺07]) provide only *very weak* guarantees of access during flooding attacks as they are (at best linearly) dependent on the number of global attack sources; e.g., a large number of “bots” could still flood a chosen link despite such guarantees. More sophisticated application-layer solutions (e.g., CAPTCHA [vABHL03]) that attempt to distinguish between human- and machine-initiated traffic to prevent flooding attacks are impractical at the network-link level.

Our Contributions. The central problem addressed in Chapter 2 is that of providing *dependable* access guarantees for the *capability setup channel*, namely for initial capability requests, during flooding attacks against routers. To be meaningful, these guarantees have to be independent of the number of attack sources (i.e., the size of a global “botnet”). In the worst case, they can only depend on attack sources in *defined locales* or neighborhoods (e.g., a local administrative domain or a set of domains in the Internet). As a consequence, competing requests for a capability to a flooded link that originate outside a contaminated locale should be unaffected, or only minimally affected, by a flooding attack, and should receive strong access guarantees. In contrast, initial capability requests originating from “bot”-contaminated locales could receive weaker access guarantees, namely guaran-

tees that depend only on the number of “bots” in the contaminated locale (but not on *all* “bots” of a multi-domain attack network). In short, our notion of dependable access to a flooded link provides *differential guarantees* for the capability setup channel. Such guarantees are possible because the distribution of attack sources in the Internet are highly *non-uniform*: some domains include sufficiently strong security mechanisms that enable them to counter or deter contamination; others are easily contaminated by “bots.” Non-uniform distribution of attack sources is evident in a variety of worm propagation models [SPW02, DZL06, RMT06, RZMT06], evolutionary features of previous worms such as CodeRed I/II, Nimda and Slammer. Differential access guarantees are desirable because they provide incentives for employing host security measures within administrative domains that prevent “botnet” (and other malware) contamination. In exchange, uncontaminated domains receive precise guarantees of link access for the *capability setup channel*, which support meaningful network-link and, ultimately, service-access guarantees.

Our scheme for providing dependable access guarantees for initial capability requests relies on three basic mechanisms. First, we define a new “*path identification*” mechanism that provides an unforgeable domain identifier to individual packets, and enables remote routers to identify a packet’s domain of origin. Second, we define a “*dynamic virtual queueing*” mechanism that guarantees a minimum number of router buffer slots to domains originating flows through a router, which in effect, guarantees link access to those domains. Finally, we employ a “*path aggregation*” mechanism that allocates router bandwidth to capability requests depending on domain contamination and maximizes service to legitimate capability requests.

1.2 Dependable Link Access for Legitimate Traffic

“Capability” schemes [ARW03, YPS04, YWA05] have aimed to provide identifier authenticity for individual flows via router-generated unforgeable identifiers (i.e., capabilities) created during the connection setup phase. These schemes effectively prevent identifier (IP address) spoofing and filter unauthorized traffic by making use of strong source-authorization rules applied at a network edge (e.g., by IDSs, Firewalls). Flow-identifier authenticity, though effective in preventing address-spoofing attacks, is insufficient to counter link-flooding attacks, for much the same reason as other per-flow defense schemes; e.g., a large “bot network” could acquire capabilities in a legitimate manner and then flood a targeted link. Such an attack would grow in strength at least linearly with the number of bots, and would have global side effects: legitimate flows could be denied access through the targeted link. Without controlling aggregate flow rates at network links, a capability scheme cannot counter link-flooding attacks. For example, application-server supplied bounds on the router bandwidth made available by a network capability to a packet flow [YWA05] cannot (and is not intended to) guarantee link access since these bounds cannot control the aggregate flow rates at that router in a “bot attack”.

In handling link-flooding attacks, most router-based methods proposed to date attempt to identify either attack flows or flow aggregates (i.e., sets of flows with defined characteristics) and penalize them in a specific manner. For example, a variety of preferential packet-dropping methods work on a per-flow ba-

sis [MFW01, FKSS01, PPP00, XG06], where a flow is identified by its source and destination IP addresses. Other methods, such as Pushback [MBF⁺02], identify the “flow aggregates” contributing to congestion, and control their rates by installing filters close to, or at, flow origins. Flow- and aggregate-based flooding defenses have complementary strengths depending on the types of attacks being launched. For example, a flow-based defense can limit the bandwidth of individual aggressive (i.e., attack) flows very effectively [XG06], yet it is ineffective for defending against large-scale attacks comprising multiple low-rate flows launched from multiple, bot-contaminated hosts – a situation in which an aggregate-based defense is needed. However, aggregate-based defenses fail to limit “collateral damage” within flow aggregates; i.e., they may deny link access to legitimate packet flows within attack aggregates [MBF⁺02, XG05, CLSS08]. In principle, integrating salient features of both types of flooding-defense mechanisms can produce practical countermeasures.

Our Contributions. In Chapter 3, we present a router-based subsystem called FLoc (Flow Localization) that confines collateral damage of link-flooding attacks within specified packet-flow “locales,” namely flows of single domains (Autonomous Systems, ASs) or specific sets of domains, and provides (1) link-access (i.e., bandwidth) guarantees on a per domain basis, and (2) fair bandwidth allocation for individual flows within a domain.

FLoc distinguishes between the flows (and flow aggregates) of bot-contaminated and uncontaminated domains. This distinction is possible for two reasons. First, the distribution of host contamination with “malware” in the Internet is highly non-uniform [SPW02, RMT06, DZL06, CG05, CJB08]: domains that employ sufficiently

strong security mechanisms and policies for individual hosts are less likely to be contaminated with malware (including attack “bots”) than others. Second, legitimate (i.e., non-attack) flows originating from uncontaminated domains have more homogeneous congestion characteristics, such as mean packet-drop rates, than flows of contaminated domains. This enables FLoc to identify attack flows of contaminated domains and confine their effects to those domains. It also enables FLoc to provide *differential bandwidth guarantees* at a congested link, in two ways: (1) uncontaminated domains receive better bandwidth guarantees than contaminated domains; and (2) legitimate flows of contaminated domains are guaranteed substantially more bandwidth than attack flows. FLoc provides differential bandwidth guarantees by two complementary rate-control policies, namely, an intra-domain preferential drop policy and inter-domain, path-identifier aggregation policy. Both are discussed in Sections 3.4.2 and 3.4.3 in detail.

1.3 Collaborative Defense for Mitigating Attack Effects

Rate limiting and filtering attack traffic are two fundamental ways of defending link-flooding DoS attacks, yet precise identification of attack flows and placement of filters determine their effectiveness. Ideally, filters are desired to be located near the origin of attack flows in order to prevent attack traffic from being injected deep into the Internet. However, such near source filtering raises some issues: on

receiving a rate-control request from a congested router/domain, a source domain (stub/provider) needs to limit its customers' traffic on others' behalf, and it cannot easily validate the legitimacy of the rate-control request (e.g., authenticity, fair bandwidth allocation).

As an alternative to “restrictive” source-end rate-control, a “promotive” routing approach can be considered in defending flooding attacks, where source domains (that originate *legitimate* flows) reroute the flows destined to a flooded link to avoid severely congested paths. It is promotive because employing rerouting offers tangible benefit (e.g., higher bandwidth, lower delay) to the flows of participating domains. However, during link-flooding attacks, traditional approaches to diversifying traffic routes, which aim for load balancing, congestion avoidance, or secure path selection, may have negative effects on legitimate traffic. For instance, multipath routing, either in the form of AS paths or intra-domain paths, would distribute attack traffic as well, thereby widens the effects of attacks. Worse yet, route diversity could disturb identification and throttling attempts of attack traffic at remote routers. Accordingly, it is necessary for routers to exploit the route diversity of the Internet exclusively for delivering legitimate traffic.

Our Contributions. In Chapter 4, we present a mutually-controlled path diversification mechanism, where administrative domains (i.e., ASs) exploit the path diversity of the Internet for routing legitimate traffic, while complying to the routing policies (i.e., BGP import/export policies) of other domains. Meanwhile, suspicious flows¹ of flooding attacks are routed through a default path that is nailed down to

¹We define potential but non-verifiable attack flows as suspicious flows.

minimize their impact on other legitimate flows – attack localization. In summary, we mitigate and localize the effects of link flooding attacks by applying “differential” routing policies. The constituting mechanisms for achieving this goal are:

- *Mutually-Controlled Multipath Routing.* Path diversity in the Internet is exploited in a mutually controlled manner by the domains at network edges, leaving the core intact: a congested router informs a source/provider domain of its congestion and a preferred domain(s) through which packets are delivered, and the notified source/provider domain selects the best path (AS path) that includes the designated domain. Of course, only legitimate domains (distinguished as such at the congested router) would take the advantage of path diversity informed by the congested router, while end-hosts (potential attack sources) have no control over path selection.
- *Path Pinning.* While routers reroute flows that originate from uncontaminated domains through less congested paths, they forward attack flows that originate from contaminated domains through a single, default path to confine attack effects on other legitimate flows. To this end, a congested router sends a path pinning request for attack flows back to the upstream domains, and the requested domains suppress route changes on those flows. Thus, attack flows would not take advantage of multiple/alternate path routing supported either by AS path diversity or by IGP’s best path selection process.
- *Rate Throttling.* Throttling the bandwidth of attack flows near their origin, if possible, can best counter flooding attacks. To motivate source-end defense,

the bandwidth of flooded link is differentially allocated to source domains based on their compliance with the rate-throttling request by the congested router – reward policy.

We design a complementary routing system equipped with the above features under the following considerations:

- **Deployment Incentive.** Multipath routing enables source domains to direct the flows that would experience severe congestion via a less congested path, and packet marking and bandwidth control to prioritize traffic originating from them (which otherwise would be equally treated at the congested link) based on customer/provider relationship. Hence, both mechanisms provide positive incentive to (source) domains that employ our scheme.
- **Policy Compliance and Backward Compatibility.** Our multipath routing operates based on the (multiple) paths exposed by other domains. Thus, it conforms to the route export policies applied at downstream domains (through BGP). And, by exploiting path diversity in an AS level, our scheme discloses neither intra-domain topology nor route import/export policy of a domain to other domains.
- **Little Overhead.** We implement rerouting by applying import policies to the border routers of a domain. This requires an additional control message from a route controller to the border routers (see section 4.4.1), yet would not cause extra overhead to control plane as well as data plane. While path

pinning requires an extra functionality (i.e., tunneling), routers that perform path-pinning trade off authenticity validation and tunnel-identifier overloading in the packet (using a capability) with forwarding table lookup. This, in effect, prevents unauthorized flows at their destination from wasting the bandwidth of the requested domains.

1.4 Organization

The balance of this dissertation is organized as follows. In Chapter 2, we present a scheme that provides link access guarantees for capability requests. We then provide bandwidth guarantees to the legitimate flows by identifying and rate-limiting attack flows in Chapter 3. In Chapter 4, we present a route control scheme that mitigates the effects of flooding attacks on the legitimate traffic that originate from uncontaminated domains. We then conclude in Chapter 5.

Chapter 2

Dependable Connection Setup for Network Capabilities

2.1 Outline of the Chapter

We first explore related work in Section 2.2 and provide an overview of our scheme in Section 2.3. In Section 2.4, we present a new, path identification mechanism that provides each packet with an unforgeable domain identifier of its origin. Precise link-access guarantees are provided to those path-identifiers by a dynamic virtual queueing mechanism presented in Section 2.5 and those guarantees are preserved in the face of wide dispersion of attack sources by a path aggregation mechanism presented in Section 2.6. The effectiveness of our scheme is evaluated by ns2 simulations under different attack scenarios and by comparing the results of these simulations with those of TVA [YWA08] in Section 2.7.

2.2 Background and Related Work

Lack of source address authenticity in the Internet Protocol (IP) enables attackers to forge the source addresses, and hence makes address-based accounting difficult

during link flooding attacks. As a way to add authenticity to a packet, capability solutions [ARW03, YPS04, YWA05, YWA08] have been proposed. Generally, a network-layer capability protocol requires a handshake between a client and a server, and during that phase, routers on the forwarding path collectively issue a connection capability; i.e., a series of router capabilities on the path. This capability protocol can be described as follows.

Capability Protocol. During the connection establishment phase of a flow, a router R_j generates a capability for that connection by hashing the packet’s source and destination addresses (IP_S, IP_D) with the router’s secret (K_{R_j}); i.e., $\text{Hash}(IP_S, IP_D, K_{R_j})$. The router writes the capability in the client’s connection request packet (e.g., TCP-SYN) to the server. Thus, a router issues an authenticated identifier for a flow that can be verified only by the router itself. This capability is returned to the client along with the server’s acknowledgement (e.g., TCP-SYN/ACK), and is carried in the client’s subsequent packets to the server. In Figure 2.1, the flow identifier f_i at R_j is the capability C_j since C_j can only be authenticated by the R_j . In this way, the flow authenticity is guaranteed at every router.

However, as pointed out in [AC05b], the capability request protocol is still vulnerable to flooding (DoC) attacks. That is, flooding with capability requests, which cannot be prioritized, successfully denies a legitimate access to a congested link. Portcullis [PWS⁺07] proposes a puzzle based mechanism that provides a guaranteed link access during a flooding (DoC) attack. Though useful, any guarantee that depends on the client’s computational power and the number of attack sources could be weakened if adversaries compromise a large number of “bots” in the In-

ternet (e.g., the size of a botnet easily exceeds 1 million bots [bot]). Alternatively, TVA’s implementation of fair queueing on incoming traffic paths (i.e., hierarchical fair queueing) [YWA08], which equally assigns a queue to each of directly connected links and recursively splits the queue for distant links, places legitimate accesses of remote domains at a significant disadvantage since it provides fair service to the same level of queues (i.e., sub-queues split from a queue).

Attempts to block suspicious traffic upstream of a congested router by installing filters close to, or at, the domains originating attacks could protect legitimate flows that are independent of attacks (e.g., different destination prefixes) [MBF⁺02]. To be effective, cooperative filtering would require incentives that scale with the number of participating domains – a tall order since it depends on the attack itself. Furthermore, with only local information (the traffic rate of incoming links), a router cannot easily identify the links (or upstream links) that are responsible for the congestion; and even if such information is available, an adversary can launch a timed attack where different groups of zombies/bots issue targeted requests by exploiting the time delay required for installing and releasing filters at upstream routers (e.g., on-off and rolling attacks).

2.3 Design Overview

In this section, we present an overview of our defense scheme by motivating constituting mechanisms.

2.3.1 Threats

The main threat we deal with in this work is the link flooding attacks on the capability-setup channel, where attack sources collaboratively exhaust the link bandwidth assigned for connection establishment. We assume that both hosts and routers can be compromised and send/forward attack traffic. Compromised hosts flood a target link with capability request packets and disturb the path identification mechanism at a remote router by manipulating the header reserved for that purpose (viz., Section 2.4.1). Whereas, compromised routers only disturb path identification by either bypassing false path-markings or adding invalid path markings to the packets they forward without actively generating attack packets.

2.3.2 Path Identification

In identifying the source domain of a packet, we use the packet’s routing path from its origin to destination to take advantage of two features provided by it. First, a packet’s routing path, when constructed by the routers on the packet forwarding path like previous path identification approaches [YPS03, YWA08], can be used as an authentic (meaning unspoofable) identifier, because it cannot be controlled by the end-hosts¹. Second, routing paths can locate packets’ origins in the Internet by enabling a remote router to construct a traffic tree. The domain connectivity revealed in the traffic tree helps identify the distribution of attack sources in specified

¹IP source routing may allow a client to select a path to a destination. However, loose source routing is usually blocked at routers to avoid spoofing attacks, and strict source routing cannot be easily available at an end-host since there is no easy way to construct an exact routing path to the destination at the end host.

locales to which bandwidth allocation will be optimized (viz., Section 2.6).

The basic concept of route construction is similar with that of previous schemes [YPS03, YWA08], yet we use a packet’s AS (Autonomous System) path as a domain identifier for a couple of reasons. First, a packet’s AS path, which is primarily determined by the number of AS hops (AS path length) to the destination in the inter-domain routing protocol (e.g., BGP-4 [RL95]), is more stable than its full routing path that may frequently change during flooding attacks due to link state changes (e.g., link failure). Namely, the AS path of a packet could be used as a persistent domain identifier. Second, a packet’s AS path can be constructed by the egress router of the source domain that contains the AS path information of destination addresses in its routing table. This source-constructible domain identifier eliminates deployment issues that previous path-marking schemes have especially at the Internet core, and hence enables independent adoption of the (path) marking scheme at the Internet border (e.g., provider/stub domains). We envision that prioritizing requests originating from path-marking domains would encourage early adoption of the marking scheme.

We define a packet’s AS path to its destination as the “path-identifier” of the packet, and present it in the order of marking: from the origin to the destination. Thus, as illustrated in Figure 2.1, the path-identifier seen at a congested router in AS_1 is $\{AS_4, AS_3, AS_2, AS_1\}$. We implement the path-identifier in a shim header like capabilities. Throughout this paper, we denote the path-identifier whose marking starts with AS_i by “ S_i ” and the BGP speaker of AS_i by “ R_i ”. In Section 2.4, we present a mechanism that protects path-identifiers from potential attacks.

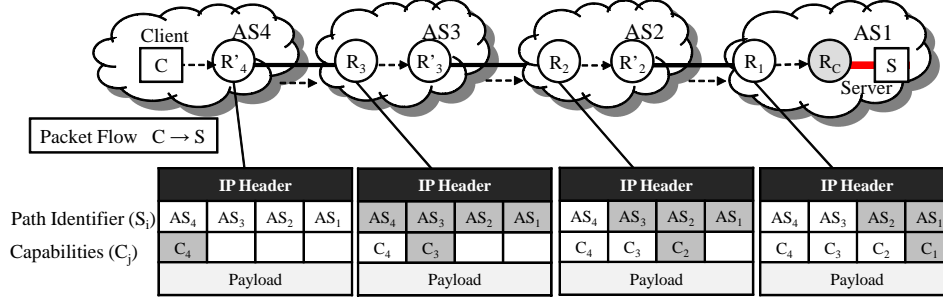


Figure 2.1: Path Identifier. R'_4 is the egress router of AS4 and R_3, R_2, R_1 are the ingress routers of AS3, AS2, AS1 respectively. R_C stands for the congested router. R'_4 writes the path-identifier to the packet heading to server S in AS1, and ingress routers on the path validate the marking. C_j is the capability issued by R_j . Each ingress router can validate the shaded part of the marking.

2.3.3 Link Access Guarantees

In defending flooding attacks on capability-setup channel, our goal is to provide precise guarantees of link access to the capability requests, where the guarantees are provided in a domain basis to confine the effects of attacks within the domains they originate. This goal is achieved by a new fair queueing mechanism, and the guarantees provided by the queueing mechanism are optimized to favor the requests from uncontaminated domains via a path aggregation mechanism.

2.3.3.1 Fair Queueing Revisited

The choice of a fair queueing scheme for link-access guarantees is intended to maximize service on the legitimate capability requests. Fair queueing schemes, if they can assign distinct queues to path-identifiers, could provide fair bandwidth to the path-identifiers without link under-utilization (which could occur whenever strict bandwidth reservation is made to individual path-identifiers). However, when the spatio-temporal dynamics of domains contributing to congestion (e.g., time-varying

patterns of domain traffic) are considered, such queue assignment in a limited buffer is also a challenging problem. For example, for a fixed buffer size, under-provisioning of the number of queues in a specific time period may fail to provide link-access guarantees on path-identifiers due to potential queue collisions among different path-identifiers. In contrast, over-provisioning of it would decrease the size of individual queues, hence weaken the guarantees (viz., Section 2.5). Thus, we aim to design a fair queueing scheme that assigns a unique queue to each path-identifier and adjusts the individual queue sizes to fit the buffer size for link-access guarantees and their enhancement – a desired goal.

While a variety of traditional fair queueing schemes focus on the bandwidth fairness of flows in different queues that contain various sizes of packets, the “Stochastic Fair Queueing (SFQ)” scheme [Pau90] offers queue length fairness through a “buffer stealing” mechanism, whereby a packet that finds buffer full on its arrival would steal a buffer-slot from the longest queue. We note that the “fixed size” capability request packet would eliminate the intrinsic bandwidth unfairness of SFQ in the presence of different packet sizes [Shr95]. Based on the buffer-stealing idea, we improve SFQ in two respects. First, we avoid queue collisions among path-identifiers that are allowed but fairly distributed via stochastic queue assignment in SFQ. Second, we make queue management operations (e.g., queue assignment and buffer-slot preemption) scalable and efficient to easily adapt our scheme to diverse operating environments (e.g., link capacity, the number of required queues). Those improvements are made by a dynamic virtual queueing mechanism below.

2.3.3.2 Path Aggregation

As more domains are contaminated by attack sources, link-access guarantees provided by our queueing scheme become weak as both available link bandwidth and buffer-slots to each path-identifier decrease. This undesirable dependency of guarantees on attack dispersion is unavoidable as long as all path-identifiers are “equally” treated. Protecting requests of uncontaminated domains essentially needs a differential treatment of path-identifiers based on the proportion of legitimate requests they deliver. Though the legitimacy of individual capability requests cannot be validated, the proportion of legitimate requests in a set of requests can be estimated by a couple of flow conformance tests, which consist of (1) a test on “bandwidth conformance” that represents the aggressiveness of requests and (2) a test on “protocol conformance” that indicates the legitimacy of authorized flows in various respects (viz., Section 2.6.1).

Conformance tests performed on each path-identifier enables differential assignment of bandwidth to path-identifiers that maximizes service to legitimate requests at the flooded link. Yet, in the presence of large number of attack domains, such assignment cannot be easily made, nor it tolerate imprecise measurement of domain contamination. Instead, we aggregate the path-identifiers of a highly contaminated locale and assign a new path-identifier to them. This, in effect, limits both available bandwidth and buffer space for those path-identifiers.

Besides the technical ease in offering differential guarantees, aggregations based on the path-identifier prefix would (1) confine the attack effects to the nearby

domains whose traffic can possibly be policed by a single administrative authority (e.g., provider/in-transit AS), and (2) make further optimizations viable, e.g., installing filter at the rendezvous point of the aggregated paths (e.g., pushback [MBF⁺02,IB02]) or path bandwidth control for authorized flows². We mathematically formulate this aggregation problem as a constrained optimization problem, and provides an efficient algorithm in Section 2.6.3.

2.4 Path Identification

In this section, we start with the basic path identification mechanism, and then enhance the mechanism with additional security features.

The basic path identification mechanism works as follows. When the egress router of a domain (i.e., the BGP speaker) forwards a packet that originates from its domain, it writes the path-identifier (i.e., the AS path to the destination) in the packet's header. AS ingress routers of the packet forwarding path can validate the authenticity of a fraction of this path-identifier starting with the upstream AS that forwarded the packet and ending with the destination AS as shown in Figure 2.1. Whenever AS ingress routers receive a non-marked packet, they write their own path-markings: the AS path from their upstream AS to the destination AS. Thus, packets that originate from a marking domain have a unique path-identifier, yet packets from a non-marking domain could share the same path-identifier with those from different domains.

²Aggregating flows that have heterogeneous path characteristics (e.g., delay) would deteriorate fair bandwidth utilization among those flows or make any such effort difficult.

As remote domains can validate only a part of path markings, attack sources in unprotected (non-marking) domains may spoof path-identifiers unless the marking scheme (which includes the verification function) is sufficiently deployed. Even under wide deployment of the marking scheme, the authenticity of path-identifiers verified at a domain cannot be delegated to the downstream domains without a strong trust relationship established between those domains. This makes any manipulation of path-identifiers by compromised routers undetectable at remote routers. To protect path-identifiers from potential attacks (e.g., spoofing and replay attacks), we present a “secure” path identification mechanism below.

2.4.1 Unspoofable path-identifier

We first introduce potential attacks that disturb path-identification at remote routers and present our defense mechanism against those attacks.

Spoofing Attack: Let $\{AS_n, \dots, AS_2, AS_1\}$ be the path-identifier seen at the congested router, and let $*$ and $\#$ denote any valid and forged sequence of markings respectively.

- Spoofing by Sources: Compromised sources in unprotected domains (by our marking scheme) can forge a path-identifier as $\{\#, AS_i, *, AS_1\}$ if domains up to AS_i are unprotected.
- Spoofing by Routers: Compromised routers in AS_k can forge a path-identifier as $\{\#, AS_k, *, AS_1\}$, or as $\{\#, AS_i, *, AS_1\}$ if domains from AS_{k-1} to AS_i is unprotected (where $k - 1 \geq i$).

In principle, a router can authenticate its path-markings by adding a digital signature on the marking. However, adding a digital signature in every packet would impose significant computational overhead for both its generation and verification. Moreover, a per-packet signature, if employed, could be exploited by attackers to exhaust routers' computational resources (e.g., by flooding small size packets). Instead, we present an efficient path-identifier authentication mechanism (in the sense that authentication does not require per-packet cryptographic operation), where each domain pre-distributes its domain-authenticator and uses it to authenticate its path-markings. One fundamental assumption for implementing this mechanism is that any protected AS has a public-private key pair certified by a trusted certificate authority (e.g., ICANN).

2.4.1.1 Authenticator Distribution

When a BGP speaker advertises an address prefix that belongs to its domain, the BGP speaker adds an origin authentication number (OAN), which is unique in its domain and is digitally signed with the domain's private-key, to its route advertisement. This can be implemented along with a secure BGP mechanism (e.g., sBGP [Ste00],soBGP [Rus03]) as it employs a digital signature scheme in advertising routes for address and route attestation purposes. All domains that receive this route advertisement hold the authenticated ASN(AS Number)-OAN pair of the origin in their routing table for later path-identifier authentication.

2.4.1.2 Origin Authentication

The BGP speaker of a packet’s domain of origin writes its ASN-OAN pair followed by the AS path to the destination in the path-identifier header. Figure 2.2 illustrates the cases for origin authentication under different deployment scenarios of the marking scheme; e.g., a marking origin AS in ① and a non-marking origin AS in ②. Whenever no path-identifier is present in a packet, the ingress router of a marking AS constructs path-markings with its own ASN-OAN pair (viz., ② in Figure 2.2). Invalid path-markings can be identified even in the presence of consecutive non-marking ASes on the paths and be filtered on the way to, or at the destination AS.

Meanwhile, a compromised router in AS_i can forge two types of valid path-identifiers such as $\{AS_i, OAN_i^k, *\}$ and $\{\#, AS_i, OAN_i^k, *\}$. However, their effects can be limited to at most those of two path-identifiers by aggregating (i.e., discarding) the non-authenticated prefixes of path-identifiers.

2.4.2 Preventing Replay Attacks

Under partial deployment of our path-marking scheme, attack sources in unprotected domains may forge path-identifiers ending with authenticated ASN-OAN pairs (since ASN-OAN pairs are not confidential to end-hosts) and use them in flooding a target link. That is, authenticated ASN-OAN pairs can be replayed. Such “replay” attacks would significantly affect the requests of protected domains.

Path-marking routers counter replay attacks via fast OAN renewals, which are efficiently implemented using a reverse hash chain [PWS⁺07]. Let OAN_i^o be the

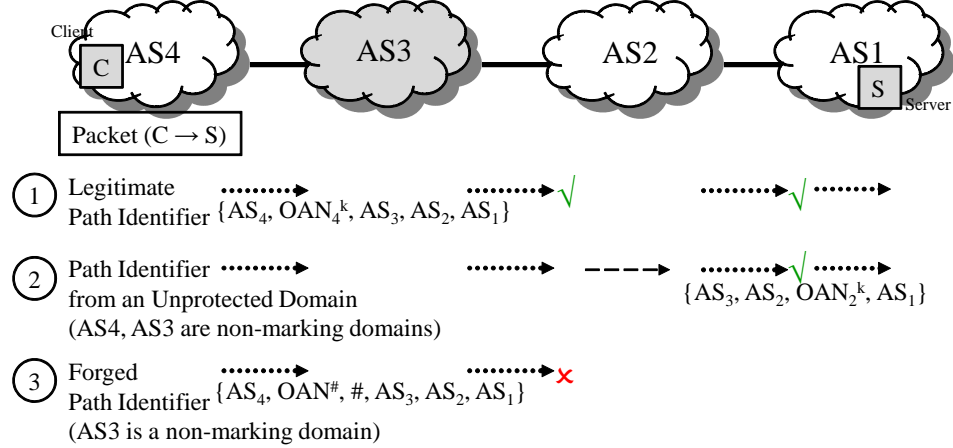


Figure 2.2: Path-identifier Authentication. ① Path-identifier written at the packet’s origin (AS4) can be validated at any domain (e.g., AS2, AS1) in the presence of a non-marking domain(s) (e.g., AS3) on the packet’s forwarding path. ② If the origin AS (e.g., AS4) does not participating in the marking, the first participant on the path (e.g., AS2) writes its marking and adds the incoming AS number (AS3) to distinguish the packets it forwards from the ones originating from it. ③ An invalid ASN-OAN pair (e.g., $OAN^\#$ represents a forged OAN at AS4) can be detected and filtered at any participating domain.

initial OAN of AS_i . AS_i constructs a hash chain of OANs by repeatedly hashing OAN_i^0 with a cryptographic hash function (i.e., $OAN_i^k = \text{Hash}(OAN_i^{k-1} || AS_i || k - 1)$ for $1 \leq k \leq M$), and distributes OAN_i^M (with M) when advertising a route. We engage AS_i and $k - 1$ in generating OAN to produce distinct OAN sequences for different initial OANs and ASes respectively. A BGP speaker uses OAN_i^k (with k) for path-identifier marking during a predefined interval; and changes it to OAN_i^{k-1} in the next interval. Hence, without breaking the hash function, an attacker cannot construct the valid sequence of OAN_i^k s to be used. A (ingress) router can authenticate OAN_i^k by computing $\text{Hash}(OAN_i^k || AS_i || k)$ and comparing it with OAN_i^{k+1} . This OAN authentication is performed only once for every OAN renewal. Once OAN_i^k is used, OAN_i^{k+1} is invalidated. Note that if the OAN renewal period is

less than the time required for replaying OANs, replay attacks will be effectively prevented.

2.5 Dynamic Virtual Queueing

In this section, we describe a dynamic virtual queueing mechanism for link-access guarantees on path-identifiers. Our dynamic virtual queueing mechanism is designed to assign a separate queue to active path-identifiers and provide queue length fairness to the path-identifiers in a *min-max* manner. For these purposes, a router manages virtual queues rather than physically separate queues, that are distinguished by the path-identifier (S_i), its count at time t ($N_{S_i}(t)$) and packet location (memory address) (A_{S_i}) in the buffer; i.e., $(S_i, N_{S_i}(t), A_{S_i})$. Given those tuples and the buffer size L_Q , queue-length fairness on path-identifiers ($\min \max_{S_i \in \mathcal{S}} N_{S_i}(t)$ for $\sum_{S_i \in \mathcal{S}} N_{S_i}(t) = L_Q$) can be described by the following buffer-slot preemption policy. If a packet finds the buffer full on its arrival, it preempts a buffer-slot from the longest virtual queue. If the arrived packet belongs to the longest virtual queue, or its preemption produces another longest virtual queue, it would be dropped. This preemption policy provides min-max fairness in terms of the virtual queue length, hence ensures guaranteed buffer-slots to each path-identifier if the number of buffered path-identifiers can be bounded. We assume that the number of buffered path-identifiers can be statistically or deterministically bounded by $|\mathcal{S}|_{max}$ at a router (i.e., the minimum bandwidth to a legitimate path-identifier can be determined).

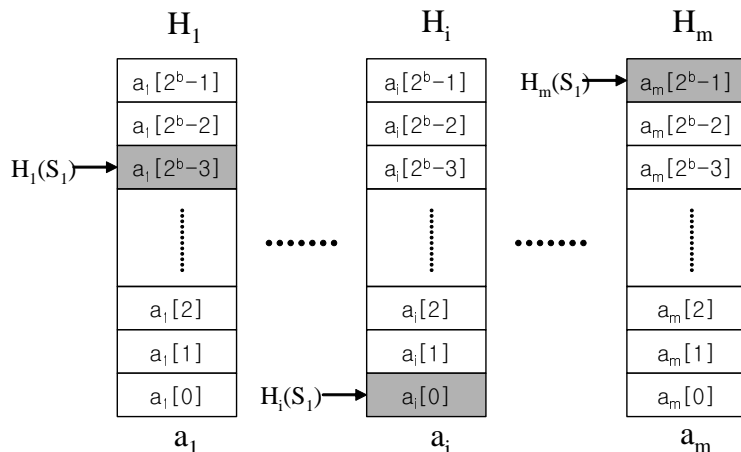


Figure 2.3: Counting Bloom Filter : In CBF, m counter arrays (a_1, \dots, a_m) are associated with m hash functions ($H_1(\cdot), \dots, H_m(\cdot)$). The counter value of S_1 in a_i is referenced by $a_i[H_i(S_1)]$.

2.5.1 Implementing Buffer-slot Preemption

For efficient and scalable accounting of virtual queue lengths, we use a new Counting Bloom Filter (CBF) that holds the number of buffer-slots occupied by path-identifiers and provides lookup, add and remove operation in $O(1)$ time (a modified version of CBF [Li 98]). CBF consists of m counter arrays of size 2^b (a_1, a_2, \dots, a_m) and m hash functions of b -bit output (H_1, H_2, \dots, H_m), where a_i is associated with H_i . For an input to CBF (e.g., S_1), each hash function maps its output to locate the corresponding array position as illustrated in Figure 2.3; i.e., $a_i[H_i(S_1)]$ corresponds to the input S_1 for $1 \leq i \leq m$.

Path-identifier accounting in CBF works as follows. All array values are initialized to zero. When a packet is added to the buffer, its path-identifier is fed into CBF. Then, CBF locates m array positions for the path-identifier, and increases the corresponding array values. The same applies to a packet removal from the buffer,

yet the counter values are decreased. In this scheme, the limited hash output size (i.e., 2^b) could cause hash collisions among path-identifiers (i.e., same hash output for different path-identifiers). Such hash collisions would make corresponding array values increased by multiple path-identifiers, hence make them corrupted. However, unless all of the array values associated with a path-identifier S_i are corrupted, we can compute the count of buffered S_i 's by taking the minimum of the array values; i.e., $\min\{a_1[H_1(S_i)], a_2[H_2(S_i)], \dots, a_m[H_m(S_i)]\}$. Since the probability that all m array values of a path-identifier are corrupted is $(1 - (1 - (1/2^b))^{|S|})^m$ for $|S|$ buffered path-identifiers [FKSS01], we can make the probability of false counting negligible by increasing the size of arrays (2^b) or the number of arrays (m).

Path-identifiers that occupy more buffer slots than the guaranteed amount (i.e., $\lfloor \frac{L_Q}{|S|} \rfloor$) should be kept track of for possible preemption. To this end, a router maintains a table, named Path-Identifier Record (PIR), that holds “over-buffered” path-identifiers, their counts and corresponding packet locations. In PIR, path-identifiers are stored as a form of signature (i.e., path-signature), where the “*path-signature*” is defined as the concatenation of m hash outputs of a path-identifier. This makes a buffer-slot preemption efficient because the preempted packet’s path-signature in PIR would directly locate array values that need to be decreased in CBF.

2.5.2 Probabilistic Guarantees

If the packet arrivals of path-identifier S_i are modeled as a Poisson process and k buffer-slots are allocated to S_i , the probabilistic lower bound of S_i 's link access (denoted by $\mathcal{G}(|\mathcal{S}|, k, S_i)$) is provided as follows.

$$\mathcal{G}(|\mathcal{S}|, k, S_i) = \begin{cases} \sum_{j=0}^{k-1} \frac{(k\rho_{S_i})^j}{j!} e^{-k\rho_{S_i}} & \rho_{S_i} < 1 \\ \frac{1}{\rho_{S_i}} (1 - \mathcal{G}_{\mathcal{L}}) (1 - \sum_{j=k}^{\infty} \frac{(k\rho_{S_i})^j}{j!} e^{-k\rho_{S_i}} \binom{j}{k-1} \mathcal{G}_{\mathcal{L}}^{k-1}) & \rho_{S_i} \geq 1 \end{cases} \quad (2.1)$$

where λ_{S_i} is the request rate of S_i , $\rho_{S_i} = \frac{\lambda_{S_i} |\mathcal{S}|}{C_R}$ is the bandwidth utilization of S_i , and $\mathcal{G}_{\mathcal{L}} = \sum_{j=0}^{k-1} \frac{(k\rho_{S_i})^j}{j!} e^{-k\rho_{S_i}}$.

We justify the Poisson arrival model of capability requests with two reasons:

(1) during the “*short*” interval that the guarantees are defined (i.e., the maximum queuing delay of a router Δ_Q), the capability requests by different clients can be assumed independent; and (2) a single capability can be used for multiple “correlated” sessions that need to be established for most Web applications. Under this model, for $\rho_{S_i} < 1$, a packet carrying S_i is guaranteed to be serviced if its queue length is less than k . In contrast, for $\rho_{S_i} \geq 1$, only a fraction of requests carrying S_i (i.e., $\frac{1}{\rho_{S_i}}$) can be guaranteed to be serviced. We provide the proof of Eq. (2.1) in Section 2.8.1. The probabilistic guarantee of S_i 's link-access is obtained by setting $|\mathcal{S}| = |\mathcal{S}|_{max}$ and $k = \lfloor \frac{L_Q}{|\mathcal{S}|_{max}} \rfloor$.

2.5.3 Resource Requirements

2.5.3.1 Request Packet Buffer

A large buffer (L_Q) for capability request packets is preferable since it would not only improve the guarantees (viz., Eq. (2.1)) but also handle the requests from spontaneously created, short-lived paths. However, the size of buffer should be bounded by the maximum allowed queuing delay to avoid unnecessary retries that occur when sources timeout. For example, if we assume 0.25 second maximum queuing delay and 120B³ request packet size, a router would require 1.6 MB buffer for 1Gbps link (when 5% of link bandwidth is assigned for capability requests); and such buffer size can provide 8 guaranteed buffer slots up to 1.67K path-identifiers.

2.5.3.2 Path-Identifier Accounting

The memory requirement for CBF is determined by a target false-positive ratio. The false positive ratio of a CBF is determined by $(1 - (1 - \frac{1}{2^b})^{|\mathcal{S}|})^m \approx (1 - e^{-\frac{|\mathcal{S}|}{2^b}})^m = (1 - e^{-\frac{L_Q}{k \cdot 2^b}})^m$ since $L_Q = k \cdot |\mathcal{S}|$. Hence, for a desired false positive ratio, the size of each counter array in CBF, which is same as the size of hash output (2^b), is linear with buffer size (i.e., $\Theta(L_Q)$). For example, 1.67K path-identifier accounting with 8 hash functions would produce a reasonably low false positive ratio ($1.58 \times 10^{-13}\%$) and require $8 \times 2\text{B}$ (hash outputs) $\times 2^8$ (counter) = 4KB for a CBF.

PIR holds the path-identifiers whose count exceeds $\lfloor \frac{L_Q}{|\mathcal{S}|} \rfloor$ for possible preemption. Hence, the memory requirement is bounded by $L_Q/(k+1) \times (32\text{B}$ (path-

³We reserve 80B shim header: 40B for path-identifier (up to 10 AS markings) and 40B for 5 capabilities.

signature) + 4B (address pointer)) (e.g., 53.4KB for the above example), since the number of path-signatures in PIR has its maximum when all path-identifiers have $k + 1$ packets in the buffer. Consequently, the memory requirement for both CBF and PIR is $\Theta(L_Q)$.

2.6 Path Aggregation

In this section, we first describe a mechanism for estimating the proportion of legitimate requests of individual path-identifiers, and then, an path-identifier aggregation mechanism that maximizes the “*goodput ratio*”, defined as the proportion of legitimate requests out of all “serviced” requests, at a congested link. As mentioned above (viz., Section 2.3.3), aggregating path-identifiers, in effect, produces an optimal traffic tree to which applying our queueing mechanism maximizes goodput ratio at the congested link.

2.6.1 Goodput Estimation

In absence of any other useful information regarding the origin of attack sources and the path-identifiers assigned to them, the request rate of path-identifier S_i (λ_{S_i}) can be used as a unique measure for estimating the goodput ratio of S_i . We define the “*bandwidth conformance*” of path-identifier S_i as $\min\{1, \frac{C_R}{\lambda_{S_i} |S|_{max}}\}$ to represent how the request rate of S_i conforms to the assigned bandwidth to it, and denote it by $\mathcal{E}_{R_i}^B$, i.e., $\mathcal{E}_{R_i}^B = \min\{1, \frac{C_R}{\lambda_{S_i} |S|_{max}}\}$ (recall that S_i is assigned to all packets originating from (or forwarded by) R_i).

Additionally, we estimate domain contamination more accurately by identifying the following attack flows: unauthorized flows, high-rate flows, and flows of high-fanout sources.

Unauthorized flows: A capability issued by a router during the connection establishment phase of a flow must be used at least once for actual data transmission unless it is denied afterward by application-layer services, firewalls or IDSs. Thus, the proportion of unused capabilities could effectively measure domain contamination much like previous schemes [MPR02,GP01], as it reflects the strong flow authorization results applied at the network ends.

High-rate flows: Flows that send high-rate traffic using valid capabilities would also exhibit high packet-drop rates as indicated in [MFW01]. Hence, if a router implements per-domain bandwidth control⁴, high-rate attack flows within a domain can be identified by capability drop rates.

High-fanout sources: If sources are allowed to establish an unlimited number of connections with other destinations through the congested link, they can deplete link’s bandwidth with a large number of “legitimate-looking” flows [SP09]. This insidious attack will be prevented if a router issues only a limited number of capabilities to a single source as follows.

Let $C_{f_{s,d}}$ be the capability for a flow $f_{s,d}$ between a source s and a destination d . $C_{f_{s,d}}$ consists of two parts, namely $C_{f_{s,d}} = C_{f_{s,d}}^0 || C_{f_{s,d}}^1$. Here, $C_{f_{s,d}}^k$ for $k \in \{0,1\}$

⁴Flows in different domains could exhibit different drops rates due to different RTTs.

is defined as:

$$\begin{aligned} C_{f_s,d}^0 &= \text{Hash}(\text{IP}_s, \text{IP}_d, K_R^1) \\ C_{f_s,d}^1 &= \text{Hash}(\text{IP}_s, f(\text{IP}_d), K_R^2) \end{aligned}$$

where IP_s and IP_d are the source and destination IP addresses, K_R^0 and K_R^1 are the router's secret keys, and $f(\cdot)$ is a function whose output is randomly uniform on $[0, n_{max}-1]$ (e.g., a universal hash function [CW77]).

$C_{f_s,d}^0$ provides identifier authenticity to flows like previous schemes [YPS04, YWA08], and $C_{f_s,d}^1$ restricts the number of per-source capabilities to n_{max} by taking $f(\text{IP}_d)$ as a hash input. If $C_{f_s,d}^1$ is used for estimating flow bandwidth, flows originating from high-fanout sources would be identified as high-rate flows.

The above attack-flow identification measures help estimate the proportion of “legitimate” flows in flows carrying S_i , which we define the “*protocol conformance*” of a path-identifier S_i and denote by $\mathcal{E}_{R_i}^{\mathcal{P}}$.

Based on the bandwidth and protocol conformances, the “*conformance estimate*” \mathcal{E}_{R_i} of S_i , representing the estimate of S_i 's goodput ratio, is defined as:

$$\mathcal{E}_{R_i} = e^{-\frac{\gamma \cdot \lambda_{S_i} |S|_{max}}{C_R}} (\mathcal{E}_{R_i}^{\mathcal{B}} - \mathcal{E}_{R_i}^{\mathcal{P}}) + \mathcal{E}_{R_i}^{\mathcal{P}} \quad (2.2)$$

$$\mathcal{E}_{R_i}(t_j) = (1 - \alpha)\mathcal{E}_{R_i} + \alpha\mathcal{E}_{R_i}(t_{j-1}) \quad (2.3)$$

where γ and α are the *weighting coefficients*.

The conformance estimate of S_i is the weighted average of the bandwidth conformance and the protocol conformance, where the weighting factor exponentially favors the protocol conformance as sufficient requests have been made⁵. We determine \mathcal{E}_{R_i} at time t_j by taking the moving average of \mathcal{E}_{R_i} s, and update it once in every aggregation period (Δ_{agg} ⁶); i.e., $t_j - t_{j-1} = \Delta_{agg}$.

2.6.2 Aggregation Problem

Based on the conformance estimates of path-identifiers, a router performs path aggregation to maximize goodput at the flooded link. To this end, the congested router R_0 builds the traffic tree (\mathcal{T}_{R_0}) using the path identifiers carried in the “active” flows and decomposes it as a combination of legitimate tree ($\mathcal{T}_{R_0}^{\mathcal{L}}$) and attack tree ($\mathcal{T}_{R_0}^{\mathcal{A}}$). $\mathcal{T}_{R_0}^{\mathcal{L}}$ is constructed by removing the leaf nodes whose conformance estimate is less than a certain threshold (\mathcal{E}_{th}) from \mathcal{T}_{R_0} ; and $\mathcal{T}_{R_0}^{\mathcal{A}}$ is constructed by removing the leaf nodes present in $\mathcal{T}_{R_0}^{\mathcal{L}}$ from \mathcal{T}_{R_0} . Then, the congested router constructs a new traffic tree \mathcal{T}'_{R_0} by merging those two trees at the root (i.e., disjoint union of $\mathcal{T}_{R_0}^{\mathcal{L}}$ and $\mathcal{T}_{R_0}^{\mathcal{A}}$), and performs aggregation on \mathcal{T}'_{R_0} . Thus, legitimate paths would never be aggregated with attack paths.

The congested router starts path aggregation from neighboring domains (i.e., domains with longest suffix-matching path-identifiers) to localize attack effects; and proceeds aggregation until a desired number of path reductions are made (viz., Eq.

⁵An insufficient number of requests from a domain could significantly bias the domain’s protocol conformance; e.g., unexpected packet drops experienced by a low-rate path-identifier would result in a very low protocol conformance.

⁶ Δ_{agg} is set to a multiple of RTT (e.g., 20·RTT in our simulations).

(2.4)). This aggregation is performed with respect to the conformance estimates of paths as guarantees of link-access should not be biased by the request rates of paths. Hence, if the number of path identifiers is bounded by $|\mathcal{S}|_{max}$, the path aggregation problem is to construct an optimal tree which has $|\mathcal{S}|_{max}$ distinct paths and to which providing link-access guarantees maximizes goodput ratio at the congested link. This can be defined as a constrained optimization problem below.

Let \mathcal{R} be the set of all nodes in \mathcal{T}'_{R_0} , and \mathcal{R}_i be the set of leaf nodes of a subtree rooted at $R_i \in \mathcal{T}'_{R_0}$ (i.e., \mathcal{T}_{R_i}). And, let $\mathcal{S}^{\mathcal{L}}$ and $\mathcal{S}^{\mathcal{A}}$ be the set of legitimate and attack path-identifiers respectively. Then, the optimization problem is defined as:

$$\begin{aligned} \max O(\mathcal{T}'_{R_0}) &= \sum_{R_i \in \mathcal{R}} \frac{1}{|\mathcal{R}_i|} \sum_{R_j \in \mathcal{R}_i} \mathcal{E}_{R_j} & (2.4) \\ \text{subject to } \sum_{R_i \in \mathcal{R}} I_{R_i} &\leq |\mathcal{S}|_{max} \text{ and } \bigsqcup_{R_i \in \mathcal{R}} \mathcal{R}_i = \mathcal{R}_0 \end{aligned}$$

where I_{R_i} is the indicator function which equals 1, if paths are aggregated at R_i , and 0, otherwise. For a non-aggregated path, I_{R_i} is 1 at the leaf node. Since $\sum_{S_i \in \mathcal{S}} I_{R_i}$ is the number of path identifiers seen at R_0 , it should be bounded by $|\mathcal{S}|_{max}$.

In the above equation, aggregation at R_i decreases the total conformance estimate (i.e., $O(\mathcal{T}'_{R_0})$) by $\frac{|\mathcal{R}_i|-1}{|\mathcal{R}_i|} \sum_{R_j \in \mathcal{R}_i} \mathcal{E}_{R_j}$, which we define as the ‘‘aggregation cost’’ and denote by $C^{\mathcal{A}}(R_i)$. Since a set of nodes at which aggregating path-identifiers has the minimum (total) aggregation cost, would be a solution to the above problem, the above optimization problem can be recast as:

$$\min O'(\mathcal{T}'_{R_0}) = \sum_{\mathcal{R}_i} C^{\mathcal{A}}(R_i) = \sum_{\mathcal{R}_i} \frac{|\mathcal{R}_i| - 1}{|\mathcal{R}_i|} \sum_{R_j \in \mathcal{R}_i} \mathcal{E}_{R_j} \quad (2.5)$$

$$\text{subject to } \sum_{R_i \in \mathcal{R}} (|\mathcal{R}_i| - 1) I_{\mathcal{R}_i} \geq k \text{ and } \bigsqcup_{R_i \in \mathcal{R}} \mathcal{R}_i = \mathcal{R}_0$$

where $k = |\mathcal{S}| - |\mathcal{S}|_{max}$.

As one can see, if the set of aggregation points (routers) are fixed, the optimization problem of Eq. (2.5) is exactly same as the *0-1 knapsack problem* ($C^{\mathcal{A}}(R_i)/|\mathcal{R}_i|$ can be considered as the unit value of an element, \mathcal{R}_i as the size of an element, and k as the knapsack size.) which is known to be NP-complete. In Eq. (2.5), the set of aggregation points and the relative aggregation cost (i.e., the unit value) of a leaf node ($\frac{|\mathcal{R}_i| - 1}{|\mathcal{R}_i|} \mathcal{E}_{R_j}$, $R_j \in \mathcal{R}_i$) vary as aggregation proceeds to the root. Hence, the *0-1 knapsack problem* should be repeatedly solved as the set of aggregation points is redefined. This means that finding an optimal solution of this problem is at least as difficult as finding that of the 0-1 knapsack problem. We present an efficient algorithm for this problem below.

2.6.3 Aggregation Algorithm

Whenever aggregation is necessary (i.e., $|\mathcal{S}| > |\mathcal{S}|_{max}$), aggregation is performed as summarized in Algorithm 1 below. Let \mathcal{O} be the solution set whose elements are the nodes at which we perform aggregation, and \mathcal{C} be the candidate set whose elements consist of all nodes that could be a solution. Initially, \mathcal{O} is empty and \mathcal{C} has all intermediate (i.e., non-leaf) nodes in \mathcal{T}'_{R_0} as its elements. Then, the algorithm

works as follows. First, the node having the lowest aggregation cost in \mathcal{C} is added to \mathcal{O} . Second, a node $R_i \in \mathcal{C}$ replaces the current solution set if its aggregation cost is less than the total aggregation cost of the nodes in \mathcal{O} . Whenever a node is added to \mathcal{O} , all its descendants are excluded from both the solution and the candidate set (to avoid the case that a single path is aggregated multiple times). They are returned to \mathcal{C} when their parent node is replaced by another node. This procedure continues until the last added node to the solution set satisfies the constraint on the number of path identifiers in Eq. (2.5).

Algorithm 1 Aggregation

- 1: Set $\mathcal{O} = \emptyset$ and $\mathcal{C} = \{R_i | R_i \in \mathcal{T}'_{R_0} - \mathcal{R}_0\}$.
 - 2: Move the lowest aggregation cost node in \mathcal{C} to \mathcal{O} .
 - 3: $R_i \in \mathcal{C}$ replaces the current solution set if it satisfies the following replacement conditions:
 - $C^A(R_i) < \sum_{R_j \in \mathcal{O}} C^A(R_j)$
 - $C^A(R_i) > \max_{R_j \in \mathcal{O}} C^A(R_j)$
 - 4: Repeat steps 2 and 3 until the minimum number of required path-identifier reductions (Eq. (2.5)) is satisfied.
-

The example in Fig. 2.4 illustrates this algorithm. The parenthesized number in the table represents the number of path-identifier reduced at each step. Starting from R_1 , when R_4 is added to \mathcal{O} , the total aggregation cost of \mathcal{O} exceeds that of R_2 . Hence, R_2 replaces all others in \mathcal{O} . In the same way, when R_5 is reached (after 5th line in the table), R_3 replaces all others in \mathcal{O} . All the descendants of the aggregated node are excluded from both the solution and the candidate set, and they are returned to \mathcal{C} when their parent is de-aggregated.

Algorithm 1 is a “greedy” approximation algorithm. However, it ensures the

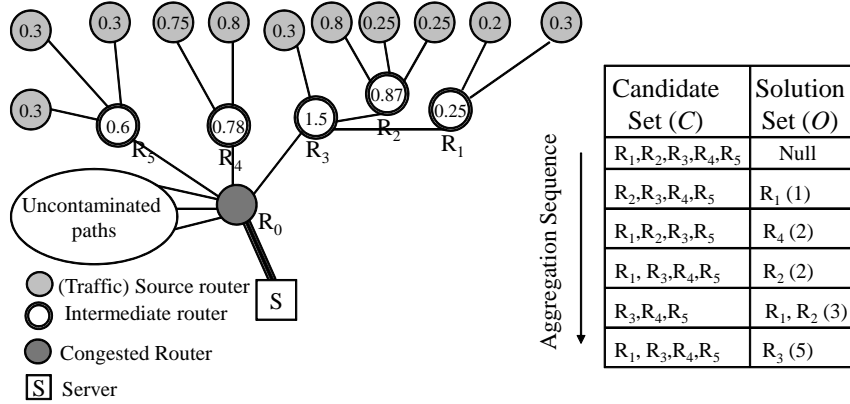


Figure 2.4: Aggregation example. The numbers in the leaf and intermediate nodes represent the conformance estimates and the aggregation costs respectively. And, the parenthesized numbers in the table represent the reduced number of path-identifiers by aggregation.

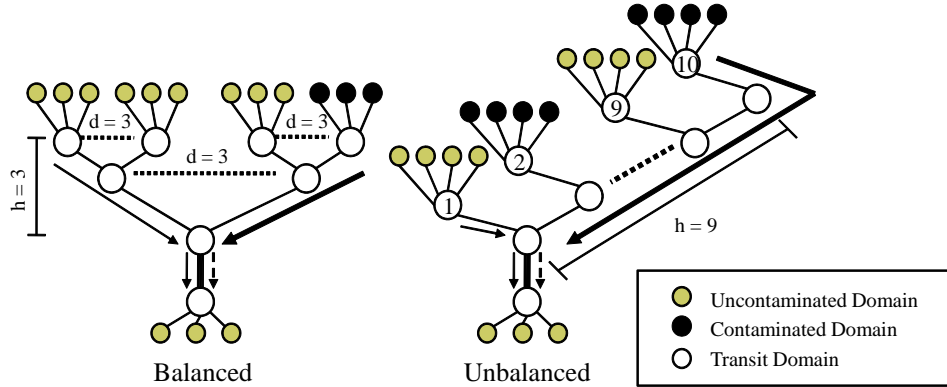


Figure 2.5: Topology used in simulation. Legend: “d” is the number of sibling nodes (degree) and “h” is the tree height.

minimum cost decreases in the candidate set, and this helps bound its distance of its solution from the optimal by the product of \mathcal{E}_{th} and the degree of the last added node. We provide the proof of this error bound in Section 2.8.2.

2.7 Simulation Results

In this section, we present our ns2 simulation results for various attack scenarios to evaluate our design. Network topologies for simulations are configured to capture the

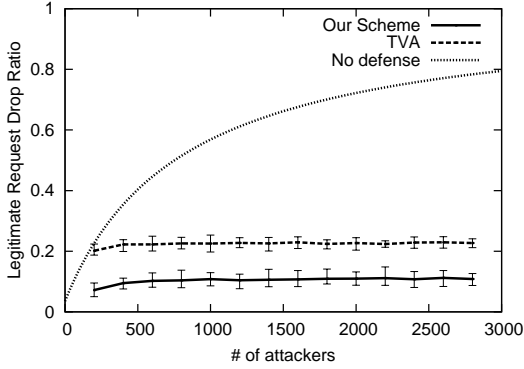


Figure 2.6: Request drop ratio of legitimate paths. Error bars represent 95% confidence intervals.

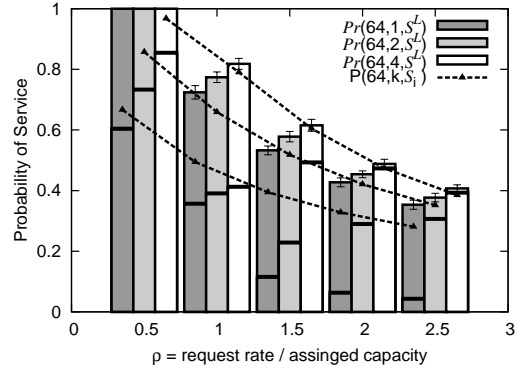


Figure 2.7: Request service probability of legitimate paths with respect to bandwidth utilization (ρ). The solid horizontal lines inside bars represent the probabilistic guarantees ($\mathcal{G}(|\mathcal{S}|, k, S_i)$).

worst case effect of different attacks and to ascertain how well our design goals are satisfied. The balanced tree shown in Figure 2.5 is used for simulations that evaluate the access guarantees and the effectiveness of aggregation. The unbalanced tree is used to show that our scheme effectively provides access guarantees to domains independently of their location on a routing path. We assign 5% of link capacity to the capability request channel as in [YWA08]. In most simulations, the total request rate of legitimate sources is set close to the link capacity of request channel (i.e., $\rho_{S_i} \approx 1$ for legitimate domains) to accurately capture the effects of attacks. Requests are randomly placed during the specified simulation interval to approximate Poisson arrivals.

We compare our simulation results with those of TVA [YWA08], which also implements a request-packet accounting scheme for the capability setup channel using a hierarchical fair-queueing algorithm.

2.7.1 Link-Access Guarantees

To evaluate the local effect of flooding attacks in our scheme, we use a 27-path balanced tree, where 30 legitimate sources are attached to each leaf node, and attack sources are increased at a leaf node. In this simulation, we set the number of access-guaranteed paths ($|\mathcal{S}|_{max}$) to 27 and the buffer size to that of 108 packets so that 4 buffer-slots are guaranteed to each path. Each source randomly starts 100 different sessions (which is equivalent to 100 times more sources) between 0 and 10 seconds. This source configuration is used for entire simulations. We also run simulations with a TVA [YWA08] router configured to have 1000 queues of length 4 (as TVA requires distinct queues for individual sources in the current implementation) for comparative evaluation.

As Figure 2.6 shows, the request drop ratios of legitimate paths are stable over the wide range of attack strengths with both our scheme and TVA. That is, both schemes effectively localize flooding attacks when compared with the “no defense” case. Note that a per-client defense would have the same result as that of “no defense” when “bots” (i.e., different machines) are used to flood the link. Yet, our scheme outperforms TVA with a much smaller buffer (108 vs. 4000 buffer-slots). This is because our scheme dynamically adjusts virtual-queue lengths in a *min-max* manner, which in effect allows more than the guaranteed buffer-slots to path-identifiers unless their bursts are synchronized (in which case, only the guaranteed buffer-slots hold).

To illustrate the robustness of the guarantees that our scheme provides, we

configure an extreme adversarial scenario where 60 paths of a 64-path balanced tree (i.e., $h = 3$ and $d = 4$ in Figure 2.5) send a large number of requests, and observe the service ratio of the remaining 4 paths. Figure 2.7 shows the probabilistic guarantee ($\mathcal{G}(|\mathcal{S}|, k, S_i)$, viz., Eq. (2.1)), the stationary service probability ($P(|\mathcal{S}|, k, S_i)$)⁷, and the simulation result ($Pr(|\mathcal{S}|, k, \mathcal{S}^{\mathcal{L}})$) under specified bandwidth utilizations – the ratio of request rate to an assigned capacity. Even under this extreme attack scenario, the service ratio of legitimate paths is close to the theoretical stationary packet service probability, which is much *higher* than the probabilistic guarantees, as illustrated in the figure.

Next, we show that link-access guarantees provided by our scheme are *independent of attack location*. For this simulation, we use a 40-path unbalanced tree (i.e., $h = 9$) as shown in Figure 2.5. We attach 30 legitimate sources to each leaf node, and 200 attack sources to each of eight attack nodes; four of these nodes are at different locations for each simulation and remaining four nodes are placed farthest from the flooded link. In this scenario, we simulate the queue implementation for $\mathcal{G}(34, 8, S_i)$, $\mathcal{G}(64, 4, S_i)$, and $\mathcal{G}(64, 8, S_i)$ and those for the corresponding 4 and 8-slot queues in a TVA router (i.e., 4000 and 8000 total buffer-slots respectively). Figure 2.8 shows the request drop ratios of legitimate paths, where the horizontal axis represents the index of attack location assigned in an ascending order of distance from the attack target (viz., unbalanced tree in Figure 2.5). With our scheme, the request drop ratios are uniform over different attack locations. This means our scheme provides

⁷For k guaranteed buffer-slots, the stationary packet service probability of S_i is determined by $P(|\mathcal{S}|, k, S_i) = 1 - \frac{\rho_{S_i}^k (1 - \rho_{S_i})}{1 - \rho_{S_i}^{k+1}}$. This is derived from the blocking probability of a M/M/1/k queueing system.

almost same protection against flooding attacks independently of the attackers' location. In contrast, TVA's performance is highly dependent upon attackers' location because TVA assigns queues to each of the directly connected links equally, splits those queues into sub-queues for distant links recursively (i.e., makes queue assignment based on a router's confidence on traffic origin), and provides fair service to the sub-queues split from a queue.

2.7.2 Aggregation Effect

Path-identifier aggregation, which optimizes domain bandwidth allocation when attack sources are widely dispersed across domains, occurs whenever the number of active paths ($|\mathcal{S}|$) becomes greater than the number of access-guaranteed paths ($|\mathcal{S}|_{max}$). In Figure 2.8, the result of the queue implementation for $\mathcal{G}(34, 8, S_i)$ illustrates the effectiveness of aggregation. As aggregation increases bandwidth allocation to legitimate paths by a factor of $\frac{|\mathcal{S}| - |\mathcal{S}|_{max}}{|\mathcal{S}|_{max}}$ (i.e., $6/34 \approx 17.6\%$ in that simulation), the request drop ratio of those paths decreases 76.8% (from 6.43% to 1.49%) when compared with that of the queue implementation for $\mathcal{G}(64, 4, S_i)$ (under which no path aggregation occurs). This is *far below* the stationary drop probability of legitimate paths (i.e., $1 - P(|\mathcal{S}|, 8, S_i) \approx 5.32\%$) that would result when physically separate queues are assigned to individual paths.

We also evaluate the effectiveness of the protocol conformance measure in aggregating attack paths. For this, we configure a 64-path balanced tree such that the same number of nodes are attached to leaf nodes to make the request rates of

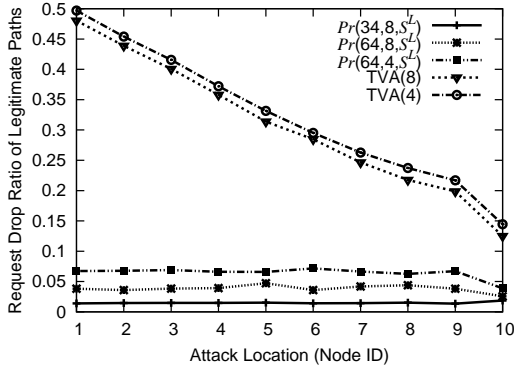


Figure 2.8: Request drop ratio of legitimate paths with respect to attack location in the unbalanced tree. TVA(k) represents the result of TVA with queue-length k.

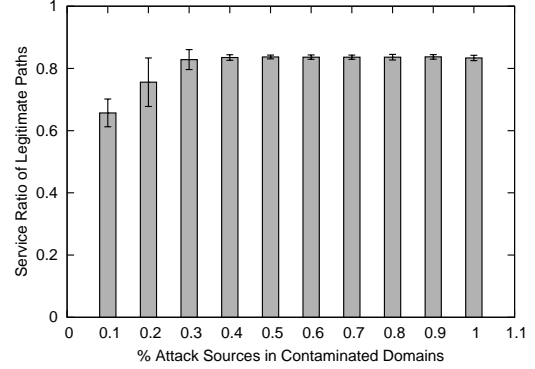


Figure 2.9: Aggregation by protocol conformance: The request service ratio of legitimate paths increases as the fraction of bots becomes higher.

all paths identical. Then, we set $|\mathcal{S}|_{max}$ to 34 (which limits the number of attack path-identifiers by at most two) and increase the fraction of attack sources whose capability requests are denied at the destination host from 10 to 100% in half of leaf nodes (i.e., 32 nodes). Note that the bandwidth conformance measure alone cannot distinguish attack paths from legitimate ones when the same request rates occur in all paths.

As Figure 2.9 shows, aggregation is more precisely performed on attack paths (which leads to higher service ratios of legitimate paths) as the fraction of attack sources in contaminated domains grows. When domains are lightly contaminated, legitimate paths can be aggregated because the cost of multi-level aggregation (i.e., aggregation at a distant node from leaf nodes) of those attack paths is higher than that of legitimate path aggregation. The high variation of service ratios for the cases that the fraction of attack sources is less than 40% is also caused by imprecise distinction between legitimate and attack paths (i.e., relatively high cost of multi-

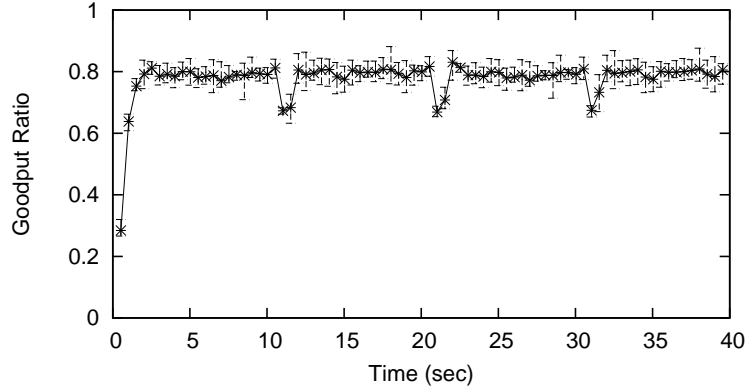


Figure 2.10: Time variation of goodput ratio at the congested link.
 Legend: Error bars represent the minimum and maximum of goodput ratio.

level aggregation).

2.7.3 Rolling Attacks

Another simulation we performed is that for the “rolling attacks”, whereby attack sources change their location to exploit delays in the response time of any defense mechanism. For this simulation, we attach 16 attack nodes at 4 different locations in the unbalanced tree (i.e., at node 1,2,9 and 10) of Figure 2.5 and place 200 attack sources in each attack node. We configure a rolling attack such that attack sources attached to node 1 and 10 flood the target for 10 seconds and the other attack sources for the next 10 seconds with a 20-second period.

In Figure 2.10, we illustrate the time variation of goodput ratio (viz., Section 2.6) at the congested link averaged over 10 runs. The goodput ratio is very low at the beginning of the simulation, since attack requests go through the target link before being preempted by legitimate ones. However, as buffer-preemption occurs (i.e., as soon as the buffer is filled) and aggregation starts (around $t =$

2), the goodput ratio rises sharply. Changing attack location in every 10 seconds significantly decreases the goodput ratio as the number of attack path-identifiers seen at the congested router increases four times (i.e., from 2 aggregated path-identifiers to 8 path-identifiers). However, these effects disappear whenever a new aggregation decision is made on the switched attack paths in Δ_{agg} (which is $20 \cdot \text{RTT} \approx 2$ seconds in this simulation).

2.8 Proof of Claims

2.8.1 Proof of Guarantees

If $\rho_{S_i} \leq 1$, a packet carrying S_i is guaranteed to be serviced if less than k arrivals of S_i have occurred in Δ_Q before its arrival. Hence, the probability of service guarantee on S_i is given as follows.

$$\begin{aligned} \Pr(\# \text{ of requests in } \Delta_Q < k) &= \sum_{j=0}^{k-1} \frac{(\lambda_{S_i} \Delta_Q)^j}{j!} e^{-\lambda_{S_i} \Delta_Q} \\ &\geq \sum_{j=0}^{k-1} \frac{(k \cdot \rho_{S_i})^j}{j!} e^{-k \cdot \rho_{S_i}} \end{aligned}$$

In contrast, for $\rho_{S_i} > 1$, a per-packet guarantee cannot be provided, since at least $\frac{\rho_{S_i}-1}{\rho_{S_i}}$ of requests must be dropped regardless of the buffer size. In this case, only a fraction of its requests can be guaranteed to be serviced (i.e., $\frac{1}{\rho_{S_i}}$), hence the probabilistic lower bound of link access is defined as the product of $\frac{1}{\rho_{S_i}}$ and

the probability that the allocated bandwidth is fully utilized. Let $P_f(S_i)$ be the probability that a packet arrival of S_i finds k buffered S_i s. The probability of full bandwidth utilization is greater than $P_f(S_i)$. Let $\mathcal{G}_{\mathcal{L}} = \sum_{j=0}^{k-1} \frac{(k \cdot \rho_{S_i})^j}{j!} e^{-k \cdot \rho_{S_i}}$. Then,

$$\begin{aligned}
P_f(S_i) &= 1 - \Pr(\# \text{ of } S_i \text{'s in the buffer} < k) \\
&\geq 1 - \left(\mathcal{G}_{\mathcal{L}} + \sum_{j=k}^{\infty} \frac{(k \cdot \rho_{S_i})^j}{j!} e^{-k \cdot \rho_{S_i}} \binom{j}{j-k+1} (1 - \mathcal{G}_{\mathcal{L}})^{j-k+1} \mathcal{G}_{\mathcal{L}}^{k-1} \right) \\
&\geq 1 - \left(\mathcal{G}_{\mathcal{L}} + \sum_{j=k}^{\infty} \frac{(k \cdot \rho_{S_i})^j}{j!} e^{-k \cdot \rho_{S_i}} \binom{j}{j-k+1} (1 - \mathcal{G}_{\mathcal{L}}) \mathcal{G}_{\mathcal{L}}^{k-1} \right) \\
&= (1 - \mathcal{G}_{\mathcal{L}}) \left(1 - \sum_{j=k}^{\infty} \frac{(k \cdot \rho_{S_i})^j}{j!} e^{-k \cdot \rho_{S_i}} \binom{j}{k-1} \mathcal{G}_{\mathcal{L}}^{k-1} \right).
\end{aligned}$$

Hence, the Eq. (2.6) follows.

2.8.2 Proof of Error Bound

We first define two types of aggregating node. In \mathcal{T}'_{R_0} , the node whose all children nodes are leaf nodes is defined as the “*leaf aggregator*” and the any other non-leaf node is defined as “*intermediate aggregator*.” The last added node to the solution set can be either a leaf aggregator or an intermediate aggregator.

If the last added node R_i to the optimal set (\mathcal{O}) is a leaf aggregator, the error from the optimal solution is bounded by $\sum_{R_j \in \mathcal{R}_i} \mathcal{E}_{R_j} \leq |\mathcal{R}_i| \cdot \mathcal{E}_{th}$, where $|\mathcal{R}_i|$ is the number of incoming links of R_i .

If the last added node to \mathcal{O} is an intermediate aggregator, we can consider two

different cases. Let R_i be an intermediate aggregator, and R_{i1}, \dots, R_{in} be the one-hop children of R_i . By the definition of aggregation cost, the following inequality can be shown.

$$C^A(R_i) = \frac{|\mathcal{R}_i| - 1}{|\mathcal{R}_i|} \sum_{R_j \in \mathcal{R}_i} \mathcal{E}_{R_j} \geq \sum_{j=1}^n C^A(R_{ij}) \quad (2.6)$$

The above inequality means that the last node added to \mathcal{O} is either (a) the node whose all immediate children aggregators are already aggregated, or (b) the node whose aggregation cost is less than the total aggregation cost of the current solution set.

case (a):

$$\begin{aligned} & C^A(R_i) - \sum_{j=1}^n C^A(R_{ij}) \\ & \leq \frac{1}{|\mathcal{R}_{i1}|} \sum_{R_j \in \mathcal{R}_{i1}} \mathcal{E}_{R_j} + \dots + \frac{1}{|\mathcal{R}_{in}|} \sum_{R_j \in \mathcal{R}_{in}} \mathcal{E}_{R_j} \\ & \leq n \cdot \mathcal{E}_{th} \end{aligned}$$

Like the leaf aggregator, if aggregation is performed at an intermediate aggregator R_i , the sum of aggregation costs of R_i 's children are deducted from the total cost. Therefore, the maximum increase of aggregation cost at an intermediate aggregator is bounded by n .

case (b):

By (2.6), a node cannot be aggregated before all of children nodes are aggregated except the case that $C^A(R_i) < \sum_{R_{oi} \in \mathcal{O}} C^A(R_{oi})$, where $\mathcal{O} = \{R_{o1}, R_{o2}, \dots, R_{on}\}$

is the optimal solution set and R_{on} is the last node added to the current solution set.

$$\begin{aligned}
C^{\mathcal{A}}(R_i) &< \sum_{R_{oi} \in \mathcal{O}} C^{\mathcal{A}}(R_{oi}) \\
\Leftrightarrow C^{\mathcal{A}}(R_i) &< \sum_{j=1}^{n-1} C^{\mathcal{A}}(R_{oj}) + C^{\mathcal{A}}(R_{on}) \\
\Leftrightarrow C^{\mathcal{A}}(R_i) - \sum_{j=1}^{n-1} C^{\mathcal{A}}(R_{oj}) &< C^{\mathcal{A}}(R_{on})
\end{aligned}$$

Hence, the increase of aggregation cost cannot be greater than the product of \mathcal{E}_{th} and the incoming-link degree of the last added node to the solution set.

Chapter 3

FLoc : Dependable Link Access for Legitimate Traffic in Flooding Attacks

3.1 Outline of the Chapter

In Section 3.2, we give an overview of previous flooding-attack defense mechanisms and discuss their weakness. Then, we present an overview of our scheme called “FLoc” in Section 3.3. In Section 3.4, we present an analytical model for dependable link access: we first describe an aggregate-flow model, and based on which, we design a scheme that provides bandwidth guarantees to aggregate-flows, identifies various types of attacks, and limits the bandwidth of the attack flows by employing preferential drop and aggregation policies. Router implementation of FLoc is presented in Section 3.5. FLoc’s effectiveness is evaluated both by simulating different attack scenarios (e.g., Constant Bit Rate (CBR) attack and Shrew [KK03]) and by comparing the FLoc results with those of other approaches (e.g., Pushback [MBF⁺02] and RED-PD [MFW01]) in the last section.

3.2 Related Work

Recent approaches to network-layer defenses against flooding attacks intend to provide authenticated identifiers to packet flows as a means of making these flows accountable. Network-layer capabilities, which were initially proposed in [ARW03] and extended in [YPS04, YWA05], help prevent address spoofing and filter out unwanted traffic at packet destinations. Another direction is to design an accountable Internet Protocol (e.g., AIP [ABF⁺08]). However, in the absence of appropriate flow control at a congested router, network links remain vulnerable to flooding attacks launched by “bot networks” using valid addresses.

Most flooding-attack defenses first identify the flows (or flow aggregates) causing link congestion and then limit their bandwidth. Traditional per-flow defense mechanisms [SSZ98, MFW01, FKSS01, PPP00] can be used in a capability-based scheme to provide legitimate flows with different types of fair bandwidth sharing. However, fair bandwidth sharing does not necessarily imply, nor is it intended to have, the ability to distinguish between legitimate and attack flows, which is a necessary requirement for achieving FLoc’s goals. For example, mechanisms that use a single packet-drop rate [MFW01], or router-queue occupancy [FKSS01, PPP00] cannot make this distinction. Using a single packet-drop rate will not work because flows, legitimate or not, can have different RTT delays and hence *different* drop rates (viz., Section 3.4.1, Eq. (3.1), where the packet drop rate is $1/(n_i T_{S_i})$) despite fair bandwidth allocation. Using router-queue occupancy will not work either, because both individual flows of attack aggregates and legitimate flows may have the *same*

router-queue occupancy for some time periods.

Neither the ability to distinguish effectively between legitimate and attack packet flows, which is necessary to identify a broad range of flooding attacks [XG05], nor fair bandwidth allocation is sufficient to confine the effects of “covert” adversary attacks (viz., Sections 3.4.2.3 and 3.6.5 for a specific example). That is, an adversary can coordinate a large number of “bots” in one contaminated domain, or more, and send only legitimate-looking (e.g., TCP-conformant, low-rate) flows through a targeted link, concurrently. This can easily deplete the bandwidth allocated (fairly) to flows of uncontaminated domains at that link [XG06, SP09]. Hence, additional mechanisms are necessary to confine the nocive effects of such attacks.

Aggregate-based countermeasures [MBF⁺02, IB02, CLSS08] mitigate the downside of per-flow defense, yet their effectiveness is highly dependent upon how flow aggregates are defined (e.g., in terms of “locales”) and how bandwidth is allocated to these aggregates. Previous approaches, such as Pushback, which install filters at remote routers, do not identify attack flow aggregates precisely since their aggregates account only for the *local* flow rates of incoming links to a router. Also, they lack effective mechanisms to limit “collateral damage” within attack aggregates and do not provide incentives to domains to perform flow filtering. Furthermore, installing filters at remote routers can be susceptible to timed attacks, whereby a “bot network” changes attack strength (e.g., on-off attacks) or location (e.g., rolling attacks) in a coordinated manner to avoid detection [XG05].

CDF-PSP [CLSS08] isolates the bandwidth of “high priority” flow aggregates, which conform to historical traffic data, from that of non-conformant “low-priority”

traffic, and limits collateral damage by allocating bandwidth proportionally to all high priority traffic first. However, CDF-PSP does not (aim to) provide bandwidth guarantees. For example, some legitimate flows that exhibit uncharacteristically high rates over a historically low-rate path receive low bandwidth allocations and, conversely, attack flows over a historically high-bandwidth path receive high bandwidth allocations. Furthermore, bandwidth isolation assumes static routing and loses effectiveness whenever dynamic traffic re-routing is caused by link congestion or failure.

3.3 Design Overview

3.3.1 Assumptions

FLoc uses the path identifier as a domain identifier (S_i) and the capability as a flow identifier (f_i). However, it does not require universal deployment since these identifiers are located outside the IP headers and only FLoc enabled routers interpret them.

3.3.2 Bandwidth Guarantees in Normal Mode

In normal mode of router operation, namely when a flooding attack is not in progress, a router assigns equal link bandwidth to all outstanding path identifiers (i.e., domains). Whenever congestion is detected, the service rate for each path identifier

is set by a separate token-bucket mechanism [Par92].¹ In principle, implementing separate token buckets for individual path identifiers could provide bandwidth guarantees to those identifiers. In practice, however, the effective bandwidth received by the aggregate flows of a specific identifier becomes highly dependent on the token-bucket parameters (i.e., on the token generation period, which determines the size of the transmission token generated at a time, and the bucket size) which are determined by the number of flows and the average RTT of the path identifier. The determination of the token bucket parameters per path identifier and that of the RTT in practice are presented in Sections 3.4.1 and 3.5.1 below.

Our token-bucket mechanism is designed to guarantee the bandwidth of persistent (long) TCP flows (e.g., FTP flows), which in effect guarantees bandwidth of short TCP flows (e.g., HTTP flows) as well. This is because short TCP flows would not experience as many consecutive packet drops (which cause congestion collapse) as those of long TCP flows, hence they are affected by flooding attacks to a lesser extent [KK03]. Additionally, our token-bucket mechanism offers features of active queue management for a path-identifier's TCP flows; e.g., early congestion notification, TCP flow de-synchronization, and fair bandwidth allocation among these flows (viz., Section 3.5).

¹The basic idea of a token bucket mechanism is that link-access tokens are generated at a constant rate and are buffered in a bucket that has a limited capacity. The token generation rate determines the guaranteed bandwidth and the bucket size specifies the maximum tolerable burst size.

3.3.3 Bandwidth Guarantees in Attack Mode

While the normal-mode, per-domain bandwidth allocation helps localize the effects of link congestion within the domains where flows originate, such localization is insufficient to counter deliberate attacks for at least two reasons. First, an attack coordinated by multiple domains reduces the bandwidth available to packet flows of legitimate domains at a targeted link. Second, the bandwidth allocated to packet flows originating from legitimate clients of highly contaminated domains is severely reduced. To handle these nocive side effects, we identify attack flows and restrict their bandwidth.

Attack-Flow Identification. We identify attack (i.e., aggressive) flows with two related mechanisms. First, we identify domains that originate attack flows. Our token-bucket mechanism allocates router bandwidth to individual domains and makes the necessary packet drops for a desired bandwidth utilization. As our token-bucket mechanism provides the reference drop rate for the flow aggregate of a legitimate domain, an excessive packet-drop rate signals the presence of attack or non-conformant flows within that domain.

Second, we identify attack flows independent of the flooding-attack strategies. To do this, we use a flow’s average packet-drop interval, defined as the “mean time to drop (MTD)” of the flow. In Section 3.4.2, we show that in FLoc, the MTDs of attack flows are distinct from those of legitimate flows, and represent the strength of attack precisely, no matter what attack strategies are employed by a “bot network”. In Section 3.5.1 we show that the MTD-based attack identification can be efficiently

implemented in a router since only the states of dropped packets need to be recorded and the lifetime of those states can be precisely bounded.

Differential Bandwidth Guarantees. Once identified, attack flows are penalized by two rate-control policies, namely preferential drop and path aggregation. These policies are designed to achieve two types of *differential bandwidth guarantees*: (1) packet flows of domains that are not contaminated by “bot networks” receive better bandwidth guarantees than flows of contaminated domains; and equally importantly, (2) within a contaminated domain, legitimate packet flows experience fewer packet drops, hence better throughput, than attack flows.

First, within a contaminated domain, the packets of the attack flows are preferentially dropped with the aim to upper bound their throughput by their fair bandwidth allocation. Accordingly, attack flows that do not respond to per-domain, fair-bandwidth controls are penalized by increasingly more packet drops. Any misidentification of legitimate flows as attack would never result in service denial once the sources of misidentified flows respond to the packet drops by decreasing their send rate. Hence, “collateral damage” (i.e., denied service) to legitimate flows within attack domains is avoided.

Second, the path identifiers of highly contaminated domains are aggregated into a single path identifier. Since router bandwidth is assigned fairly to path identifiers, aggregation, in effect, reassigns the bandwidth of highly contaminated domains to legitimate (i.e., non-contaminated or lightly contaminated) ones. Hence, path aggregation helps provide higher bandwidth guarantees to legitimate (i.e., non-aggregated) paths by reducing the bandwidth assigned to aggregated paths even

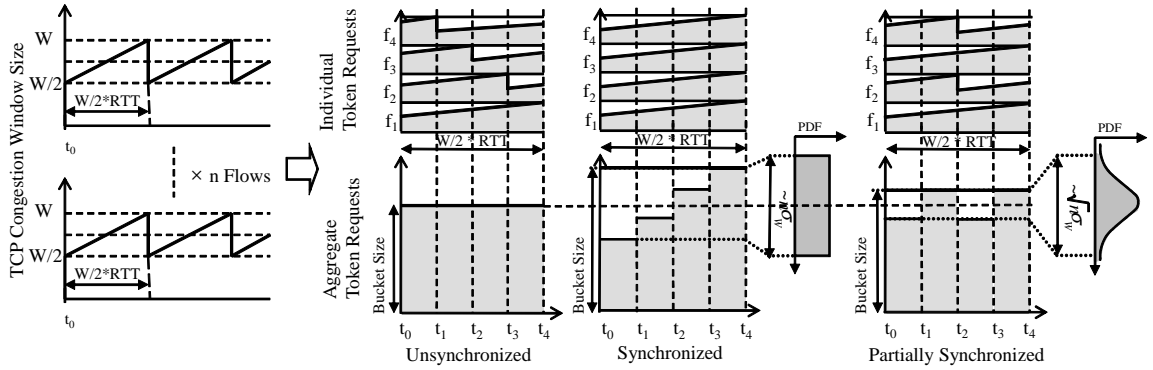


Figure 3.1: TCP window variations and their aggregate effect at a congested router for unsynchronized, partially synchronized, and synchronized flows.

Legend: Upper graphs: gray areas denote individual packet-flow rates. Lower graphs: grey areas illustrate three different distributions of the same aggregate-flow rate at a congested router.

when attack sources (e.g., bots nets) are widely dispersed across multiple domains. Aggregation is triggered whenever the number of outstanding path identifiers exceeds a limit set so that all active path identifiers receive a minimum guaranteed bandwidth at a congested router.

3.4 Modeling Dependable Link Access

In this section we present an analytical model for the bandwidth guarantees for legitimate packet flows, which define FLoc’s notion of dependable link access in flooding attacks.

3.4.1 Token-Bucket Model Revisited

In this subsection, we first review the standard TCP congestion-control mechanism and the aggregate traffic characteristics of multiple TCP flows at a congested router. Then, we define the parameters of a token bucket mechanism for a (per-domain)

path identifier that provide guaranteed bandwidth to legitimate flows in normal (non-attack) mode of router operation.

TCP Flow Model. Let W_i be the congestion window size at the source of TCP flow f_i . The TCP congestion-avoidance protocol increases the W_{f_i} of the TCP source on every acknowledgement from the destination. That is, the source increments its W_{f_i} by one in every round trip time (RTT) until it experiences a packet drop. After the packet drop, the source halves its W_{f_i} , and repeats the congestion-avoidance protocol. Figure 3.1 illustrates TCP window size variation for n flows. Window size W_{f_i} is typically modeled as a random variable whose distribution is uniform on $[\frac{W}{2}, W]$, where W is the peak congestion window size of a flow f_i 's source, as discussed in [AKM04]. In this ideal model, a source experiences a packet drop in every $W/2 \times RTT$ seconds; i.e., the “mean time to drop” for flow f_i (denoted as $MTD(f_i)$) is $MTD(f_i) = W/2 \times RTT$. Hence, for n TCP flows through a congested router, n packet drops occur during $W/2 \times RTT$ seconds if all flows share an equal bandwidth and have the same RTT.

Guaranteed Bandwidth for Legitimate Flows. To control the packet drops of TCP flows on a per-domain basis, we allocate a separate token buffer to each path identifier and customize the token generation period and bucket size for a given bandwidth guarantee. We consider three cases of TCP flow synchronization that affect traffic burstiness and hence the token bucket parameters; i.e., unsynchronized, synchronized, and partially synchronized traffic.

Let us consider the ideal (best) case, namely when the flows are completely unsynchronized; i.e., the peak window size of each source is uniformly distributed

in time, as shown in the upper graph of Figure 3.1 in the “Unsynchronized” case. Uniform distribution of individual sources’ peak window sizes makes their aggregate token-request rate at the congested router uniform over time, leading to full consumption of tokens; i.e., to full bandwidth utilization. This is illustrated in the lower graph of Figure 3.1 in the “Unsynchronized” case. Let C_{S_i} be the bandwidth guaranteed to a n_i -flow path identifier S_i , RTT_i the average RTT of the flows in S_i , and W_i the maximum window size of the flows in S_i . And, let N_{S_i} be the bucket size measured in tokens, and T_{S_i} the token generation period. To make a packet drop uniformly every $T_{S_i} = \frac{W_i/2 \times RTT_i}{n_i} = \frac{MTD(f_i)}{n_i}$ seconds, the bucket size is set to $N_{S_i} = C_{S_i} \times \frac{W_i/2 \times RTT_i}{n_i}$ tokens, and the bucket is filled within T_{S_i} seconds. That is, N_{S_i} tokens are generated at the start of each period, and the unused tokens of the previous period are removed. In this case, aggregate flows of a (per-domain) path identifier would run out of tokens only if their token requests exceed N_{S_i} in a period T_{S_i} . (Note that bursty requests are allowed within a period T_{S_i} .)

Now we can relate the window size of a flow’s source with that flow’s assigned bandwidth and RTT_i . The uniform distribution of the window size on the interval $[\frac{W_i}{2}, W_i]$ implies that the average window size is $\frac{3}{4}W_i$. Since the bandwidth C_{S_i} guaranteed to path identifier S_i is allocated fairly (i.e., divided equally) among the n_i flows of S_i , each flow’s bandwidth is C_{S_i}/n_i . Thus the relationship between a flow’s bandwidth and its window size, $bw_{f_i} = \frac{W_{f_i}}{RTT}$, implies that $C_{S_i} = \frac{3n_i W_i}{4RTT_i}$. Consequently, the token bucket parameters (i.e., token generation period and the corresponding bucket size) can be expressed as follows.

$$T_{S_i} = \frac{W_i/2 \times RTT_i}{n_i} = \frac{MTD(f_i)}{n_i} = \frac{2 C_{S_i}}{3 n_i^2} RTT_i^2 \quad (3.1)$$

$$N_{S_i} = \frac{2 C_{S_i}^2}{3 n_i^2} RTT_i^2 \quad (3.2)$$

In contrast, in the worst case, namely when all flows are completely synchronized, they would experience packet drops in the same period T_{S_i} ; viz., the interval denoted by $[t_3, t_4]$ in upper graph of Figure 3.1 in the “Synchronized” case. In this case, the peak token-request rate of aggregate flows at the congested router, that is when the window size of every flow reaches at W_i , would be the twice of its minimum that occurs when every flow halves its window size after experiencing a packet drop. Only 3/4 of the tokens generated could be consumed in this case; e.g., the shaded area of the lower graph of Figure 3.1 in the “Synchronized” case represents only 3/4 of the token generated in the interval $[t_0, t_4]$. Hence, the ideal token-bucket size needs to be increased by 1/3 to accommodate the peak flows in the worst case; i.e., it should be increased to $\frac{4}{3}N_{S_i}$.

In normal (non-attack) mode of operation, flow sources are independent (e.g., flows are not synchronized by adversaries) and they share a token-bucket’s bandwidth fairly (i.e., equally). This implies that the window size W_{f_i} of a flow f_i ’s source can be modeled as an i.i.d. random variable whose distribution is uniform on $[\frac{W_i}{2}, W_i]$ (i.e., $\mu_{W_{f_i}} = \frac{3}{4}W_i$ and $\sigma_{W_{f_i}} = \frac{1}{\sqrt{12}} \frac{W_i}{2}$). Thus, the token-request rate of a n_i TCP-flow aggregate (i.e., $\sum_{i=1}^{n_i} W_{f_i}$) has a gaussian distribution with mean $n_i \mu_{W_{f_i}}$ and standard deviation $\sqrt{n_i} \sigma_{W_{f_i}}$ (by the Central Limit Theorem). Intuitively, this can be interpreted to mean that (1) only $\sqrt{n_i}$ flows are synchronized, and (2) $\frac{1}{\sqrt{n_i}}$

subaggregate flows exist on a path. This is illustrated in the “Partially Synchronized” case of Figure 3.1. As the lower graph of this figure illustrates, partially synchronized flows do not consume all tokens of their bucket either, yet they utilize the tokens (i.e., the allocated bandwidth) better than the completely synchronized flows. This is the case because the request rates of partially synchronized flows fluctuate less than that of completely synchronized flows. This suggests that the token bucket size should be increased as a function of the flow-synchronization degree, which determines the standard deviation of a path identifier’s token-request rate.² Thus, expected traffic bursts are tolerated.

We define the new, increased size of the token bucket as $N_{S_i}^I = (1 + \frac{\epsilon\sigma_{S_i}}{\mu_{S_i}})N_{S_i}$, where μ_{S_i} and σ_{S_i} are the mean and standard deviation of the token request rate of S_i , and ϵ is the increase factor. For i.i.d flows that we consider, $\sigma_{S_i} = \sqrt{n_i}\sigma_{W_{f_i}} = \frac{\sqrt{n_i} W_i}{\sqrt{12} \cdot 2}$. We set the increase factor to $\epsilon = \sqrt{12}^3$ as this would bound the peak token requests with probability 99.97%; i.e., $\Pr(\sum_{i=1}^{n_i} W_{f_i} \leq \mu_{S_i} + \sqrt{12}\sigma_{S_i}) = 0.9997$. Accordingly, the new, increased token-bucket size for i.i.d. flows becomes

$$N_{S_i}^I = (1 + \frac{\epsilon\sqrt{n_i}\sigma_{W_{f_i}}}{n_i\mu_{W_{f_i}}})N_{S_i} = \frac{2}{3}(1 + \frac{2}{3\sqrt{n_i}})\frac{C_{S_i}^2}{n_i^2}RTT_i^2. \quad (3.3)$$

In summary, to guarantee a certain bandwidth, C_{S_i} , to a (domain) path identifier S_i in normal mode of router operation (i.e., when all flows, including bursty ones, are legitimate), we count the number of active flows, n_i , measure the average

²Note that, in general, for a given flow synchronization model (i.e., a window-size distribution), the number of additional tokens need to be provided is proportional to the standard deviation of the aggregate token request rate.

³This value is chosen under the assumption that $n \gg 4$.

RTT_i for that path identifier (viz., Section 3.5.1), and then set the token bucket parameters, namely the token generation period, T_{S_i} , and corresponding token bucket size, $N_{S_i}^T$, as specified by Eqs. (3.1) and (3.3), respectively. Note that for these token bucket parameters, the $MTD(f_i) = \frac{W_i}{2}RTT_i = n_iT_{S_i}$, as shown in the TCP Flow Model above.

3.4.2 Attack-Flow Identification and Confinement

In this subsection, we describe how to identify attack paths and attack flows within those paths by the packet drop intervals of flows, and how to limit the bandwidth of those attack flows. Let \mathcal{F}_{S_i} be a set of flows carrying path identifier S_i , and $MTD(f_i)$ be the “mean time to packet drop” of flow $f_i \in \mathcal{F}_{S_i}$. Then, $MTD(f_i)$ can be written as $MTD(f_i) = \frac{W_i}{2}RTT_i = n_iT_{S_i}$ in normal mode of operation (as shown in the previous section). We define $n_iT_{S_i}$ as the *reference* MTD of a flow carrying (domain) path identifier S_i .

3.4.2.1 Attack (Domains) Paths

For the paths that deliver attack flows (which we call “attack paths”), the MTD of aggregate flows is lower than the token generation period while the request rate of S_i (λ_{S_i}) is higher than the allocated bandwidth added by the reference drop rate of S_i ; i.e., $MTD(\mathcal{F}_{S_i}) < T_{S_i}$ and $\lambda_{S_i} > C_{S_i} + 1/T_{S_i}$. This is because the MTDs of legitimate flows is less than the reference MTD (due to the decrease of available bandwidth), yet it is greater than those of attack flows; i.e., the MTDs of all flows are less than the

reference MTD. Hence, attack paths can be identified by estimating the *mean packet drop rate* of path-identifiers, which is the inverse of MTD. If allowed, attack paths would over-utilize their bandwidth by exhausting the extra tokens made available by the new, increased bucket size (defined above). To avoid such bandwidth over-utilization, the fixed bucket size (N_{S_i}) is applied to path identifiers containing attack flows instead of the new, increased ones ($N_{S_i}^T$). This strictly limits the bandwidth available to attack paths.

3.4.2.2 Attack Flows

Though effective in localizing attack effects, bandwidth control on a (per-domain) path identifier basis does not prevent all “collateral damage;” i.e., does not protect legitimate flows that happen to be within an attack path. To protect these flows, we introduce an attack-flow identification and control mechanism.

MTD Measurement. Let $D_{S_i}^{f_i}(t_j)$ be the number of packet drops of flow f_i in interval $(t_{j-1}, t_j]$. If we set $t_j - t_{j-1} = T_{S_i}$, for some $k \geq n$, $\text{MTD}(f_i)$ under our token-based packet admission policy is measured as:

$$\text{MTD}(f_i) = \frac{k \cdot T_{S_i}}{\sum_{j=1}^k D_{S_i}^{f_i}(t_j)}. \quad (3.4)$$

Since $\text{MTD}(f_i)$ is inversely proportional to the packet-drop rate of f_i (which is proportional to its send rate), the MTD of an attack flow is *always lower* than that of a legitimate flow. This definition of MTD could identify attack flows showing vastly different drop patterns (i.e., employing different attack strategies), since it

is measured over sufficient periods kT_{S_i} (viz., Eq. (3.4)) for estimating flows' send rates.

For the attack flows identified by their MTD, the congested router applies the following packet admission policy to limit the bandwidths of those flows.

$$\Pr(f_i \text{ is serviced}) = I_{S_i}^T(f_i) \cdot \text{Min}\left\{1, \frac{\text{MTD}(f_i)}{n_i T_{S_i}}\right\} \quad (3.5)$$

where $I_{S_i}^T(f_i)$ equals 1, if a token is available to flow f_i , and 0, otherwise.

This packet admission policy preferentially drops the packets belonging to attack flows in proportion to their *send* rates, and more aggressively penalizes the flows whose MTDs keep decreasing (i.e., flows that do not respond to packet drop messages by decreasing their send rate). When an attack flow which sends traffic with rate $\alpha \frac{C_{S_i}}{n_i}$ ($\alpha > 1$) experiences d preferential packet drops, its effective bandwidth at the congested link is $\alpha \frac{C_{S_i}}{n_i} \cdot \Pr(f_i \text{ is serviced}) \leq \alpha^{1-d} \frac{C_{S_i}}{n_i}$, since $\text{MTD}(f_i)$ decreases proportionally to $\frac{1}{\alpha}$ on each preferential drop. Hence, whenever all flows actively compete for the bandwidth allocated to a path identifier (i.e., no spontaneously under-subscribing flow exists), flow f_i cannot use more bandwidth than its fair amount within that path identifier allocation. Note that the above packet admission policy would never deny service to the misidentified (attack) flows since service to those flows would resume once the sources of those flows respond to packet drops by decreasing their *send* rates; i.e., their $\text{MTD}(f_i)$ would keep increasing and so would $\Pr(f_i \text{ is serviced})$, as shown in Eq. (3.5).

3.4.2.3 Confinement of Covert Attacks

Recall that, in a “covert” attack, each “bot” may coordinate a large number of legitimate-looking (low-rate) flows to traverse a target link, concurrently. Covert attacks can be extremely potent. For example, each of N bots on one side of the targeted link can coordinate sending messages to and receiving messages from each of N “bots” on the other side of that link so as to cause $O(N^2)$ link flows [SP09]. Individually, these $O(N^2)$ flows appear to be perfectly legitimate and yet collectively they may deplete most of, if not all, that link’s bandwidth.

FLoc counters the effects of covert attacks in two ways. First, FLoc limits the number of flows that a source can make with different IP destinations through the flooded link by constructing a flow’s capability as follows.

Let $C_{f_{s,d}}$ be the capability for a flow $f_{s,d}$ between a source s and a destination d . $C_{f_{s,d}}$ consists of two parts, namely $C_{f_{s,d}} = C_{f_{s,d}}^0 || C_{f_{s,d}}^1$. Here, $C_{f_{s,d}}^k$ for $k \in \{0,1\}$ is defined as:

$$\begin{aligned} C_{f_{s,d}}^0 &= \text{Hash}(\text{IP}_s, \text{IP}_d, S_i, K_R^1) \\ C_{f_{s,d}}^1 &= \text{Hash}(\text{IP}_s, F(\text{IP}_d), S_i, K_R^2) \end{aligned}$$

where IP_s and IP_d are the source and destination IP addresses, K_R^0 and K_R^1 are the router’s secret keys, and $F(\cdot)$ is a function whose output is randomly uniform on $[0, n_{max}-1]$ (e.g., a universal hash function [CW77]). Capability $C_{f_{s,d}}^0$ provides identifier authenticity to flows [YPS04, YWA08]. Capabilities $C_{f_{s,d}}^1$ enable a router to (1)

restrict the number of flows per source to n_{max} , and (2) account for the total bandwidth requested using those capabilities at that router concurrently. Thus sources with a high fanout of legitimate, low-rate, concurrent flows would be identified as sources of high-rate, covert attack flows within a router.

Second, FLoc confines such covert attacks to individual domains by avoiding aggregation of legitimate path identifiers that have widely different numbers of flows (viz., Section 3.4.3.2). A brief analysis of FLoc’s handling of covert attacks is presented in Section 3.6.5.

3.4.3 Differential Bandwidth Guarantees

In this subsection, we present a mechanism that provides path identifiers with differential bandwidth guarantees based on a “conformance” measure for path identifiers. We define the “conformance” of a path identifier S_i ending at router R_i in a time interval $(t_{k-1}, t_k]$ as the fraction of the legitimate flows in S_i . We denoted this “path-conformance” by \mathcal{E}_{R_i} and express it as a moving average of \mathcal{E}_{R_i} values. That is,

$$\mathcal{E}_{R_i}(t_k) = \beta_{conf} \left(1 - \frac{n_i^a}{n_i}\right) + (1 - \beta_{conf}) \mathcal{E}_{R_i}(t_{k-1}) \quad (3.6)$$

where n_i and n_i^a are the number of active flows and attack flows forwarded by R_i ; and β_{conf} is a constant smoothing factor ($0 < \beta_{conf} < 1$).

Based on this path-conformance measure, a router performs (1) attack-path aggregation to maximize goodput at the flooded link and (2) legitimate-path aggre-

gation to counter potential unfair bandwidth allocation to flows of legitimate paths. For these aggregations, the congested router R_0 builds the traffic tree (\mathcal{T}_{R_0}) using the path identifiers carried in the “active” flows and decomposes it into two subtrees, namely an attack tree ($\mathcal{T}_{R_0}^A$) and a legitimate tree ($\mathcal{T}_{R_0}^L$). $\mathcal{T}_{R_0}^A$ is constructed by removing the leaf nodes whose path conformance is greater than a certain threshold (\mathcal{E}_{th}) from \mathcal{T}_{R_0} ; and $\mathcal{T}_{R_0}^L$ is constructed by removing the leaf nodes present in $\mathcal{T}_{R_0}^A$ from \mathcal{T}_{R_0} .

3.4.3.1 Attack-Path Aggregation

The goal of attack-path aggregation is to provide bandwidth guarantees to legitimate paths despite wide dispersion of attack “bots” over a large number of domains. This is achieved by aggregating the path identifiers of highly contaminated domains and hence limiting the number of bandwidth-guaranteed path identifiers. Path identifier aggregation starts from the nearby domains (i.e., domains with longest prefix-matching path identifiers) to (1) localize attack effects within these domains and (2) avoid mixing flows having highly different RTT delays (as this would affect FLoc’s precision in estimating token-bucket parameters; viz., discussion in Section 3.5.1). Whenever the number of path identifiers is bounded by $|\mathcal{S}|_{max}$, the attack-path aggregation problem can be defined as the path-conformance maximization problem below.

Let \mathcal{R} be the set of all nodes in $\mathcal{T}_{R_0}^A$, and \mathcal{R}_i be the set of leaf nodes of a subtree rooted at $R_i \in \mathcal{T}_{R_0}^A$ (i.e., $\mathcal{T}_{R_i}^A$). And, let \mathcal{S}^L and \mathcal{S}^A be the set of legitimate

and attack path-identifiers respectively. Then, the path-conformance maximization problem is defined as:

$$\max O(\mathcal{T}_{R_0}^A) = \sum_{R_i \in \mathcal{R}} \frac{1}{|\mathcal{R}_i|} \sum_{R_j \in \mathcal{R}_i} \mathcal{E}_{R_j} \quad (3.7)$$

$$\text{subject to } \sum_{R_i \in \mathcal{R}} I_{R_i} \leq |\mathcal{S}|_{max} - |\mathcal{S}^{\mathcal{L}}| \text{ and } \bigsqcup_{R_i \in \mathcal{R}} \mathcal{R}_i = \mathcal{R}_0$$

where I_{R_i} is the indicator function which equals 1, if paths are aggregated at R_i , and 0, otherwise. For a non-aggregated path, I_{R_i} is 1 at the leaf node. Since $\sum_{S_i \in \mathcal{S}} I_{R_i}$ is the total number of attack path identifiers seen at node R_0 , it should be bounded by $|\mathcal{S}|_{max} - |\mathcal{S}^{\mathcal{L}}|$ for the bandwidth guarantees on $\mathcal{S}^{\mathcal{L}}$.

In the above equation, aggregation at router R_i decreases the path-conformance by $\frac{|\mathcal{R}_i|-1}{|\mathcal{R}_i|} \sum_{R_j \in \mathcal{R}_i} \mathcal{E}_{R_j}$, which we define as the ‘‘aggregation cost’’ and denote by $C^A(R_i)$. Hence, a set of nodes at which aggregating path-identifiers has the minimum (overall) aggregation cost and reduces at least $|\mathcal{S}^A| - (|\mathcal{S}|_{max} - |\mathcal{S}^{\mathcal{L}}|)$ path-identifiers, would be a solution to the above problem. The above aggregation problem is exactly same as the problem defined for differential link-access guarantees for capability requests in Section 2.6.2 (viz., Eq. (2.4)). Hence, it is solved using Algorithm 1 presented in Section 2.6.3.

3.4.3.2 Legitimate-Path Aggregation

The aggregation of legitimate paths is intended to achieve proportional bandwidth allocation to ‘‘legitimate’’ path identifiers that have different numbers of flows; i.e.,

fair bandwidth allocation to flows. In aggregation, a (congested) router assigns a new path-identifier to the aggregated paths and allocates its bandwidth proportional to the number of aggregated paths. Consequently, the path conformance of the aggregated path would be the weighted average of individual paths' conformance measures, where the weighting factor is the number of domains' flows.

Let $C^{\mathcal{L}}(R_i)$ be the *net change* of path-conformance after the aggregation of paths at R_i . Then, $C^{\mathcal{L}}(R_i)$ is defined as:

$$C^{\mathcal{L}}(R_i) = \frac{1}{|\mathcal{R}_i|} \sum_{R_j \in \mathcal{R}_i} \mathcal{E}_{R_j} - \frac{\sum_{R_j \in \mathcal{R}_i} \mathcal{E}_{R_j} n_j}{\sum_{R_j \in \mathcal{R}_i} n_j}. \quad (3.8)$$

A negative value of $C^{\mathcal{L}}(R_i)$ means that the aggregation of paths at R_i would increase the path-conformance, and, eventually, the goodput of the flooded link. Hence, aggregation would be performed at all nodes in $\mathcal{T}_{R_0}^{\mathcal{L}}$ whose cost is negative.⁴

However, if a covert-attack path (viz., Section 3.4.2.3 and [SP09]) is inadvertently aggregated with a (truly) legitimate path, a large number of legitimate-looking flows of the covert path may soak the bandwidth [XG05] of (truly) legitimate flows. To avoid this, FLoc does not aggregate legitimate paths whenever aggregation would increase bandwidth allocation to any path by more than a fraction of its current value; e.g., in our simulation this fraction is set to 50%.⁵ Thus aggregation would never take place for path identifiers that have widely different numbers of flows.

⁴However, FLoc does not aggregate paths that have significant discrepancies in RTT delays as such aggregations would lead to false identification of attack flows.

⁵Typically, ISPs over-provision network link capacity to maintain link utilization under 40 – 50% in normal operating conditions, and under 75% in the presence of a single link failure. Hence, increasing the bandwidth of covert-attack paths by 50% or less will not exceed link capacity in practice.

3.5 Router Design

In this section, we describe how our model of dependable link-access is implemented in a router. Specifically, we describe the packet admission (and drop) policies based on the token-bucket mechanism and the management of the router’s buffer queue.

3.5.1 Token-Bucket Activation and Router Queue Management

The token bucket parameters (i.e., token generation pattern and bucket size) for a path-identifier depend quadratically on the *actual* RTT_i of a path identifier S_i (viz., Eqs. (3.1) – (3.3)). Since the actual RTT_i can only be approximated, we estimate its value by (1) averaging the measured RTT of individual flows in a path, and (2) adjusting that average downward to avoid an over-estimate (e.g., we divide the average RTT of a path by 2 in the simulations reported in the next section). Note that an over-estimate of the actual RTT_i would inflate the token generation period and bucket size substantially, thereby causing bandwidth over-utilization and overflows of the router buffer queue. In contrast, an under-estimate would deflate the token-bucket parameters and potentially cause unnecessary packet drops for a path identifier. FLoc implements a control mechanism that adjusts the packet-drop rate and compensates for any unnecessary packet drops (discussed below).

RTT_i measurement. We measure the average RTT of a path S_i at a router by averaging the individual flows’ RTTs for that path. A flow’s RTT is measured as the time between a client’s connection (i.e., capability) request and its first data

transmission. The elapsed time from a capability issue by a router to a client to the client’s first use of the issued capability at that router is a fairly accurate measurement of an individual flow’s RTT.

Router-Queue Management. The management of a router’s buffer queue establishes the service rate for path identifiers and, as such, it implements the bandwidth guarantees provided by the token-bucket mechanism for each path identifier. FLoc implements a router’s (FIFO) queue whose size varies in the interval $[Q_{min}, Q_{max}]$. Q_{min} is a configurable parameter chosen to avoid both link under-utilization (which could be caused by short bursts of packets) and long queuing delays; viz., the RED queue. (We set Q_{min} to 20% of buffer size in the simulations of Section 3.6.) To determine Q_{max} , recall that the token requests of partially synchronized flows of a path S_i oscillate between $(\frac{3n_i}{4} - \frac{\sqrt{n_i}}{2})W_i$ and $(\frac{3n_i}{4} + \frac{\sqrt{n_i}}{2})W_i$. This requires at least $\sqrt{n_i}W_i$ packet buffer space for S_i to avoid link under-utilization. Hence, we set $Q_{max} = Q_{min} + \sum_{S_i \in \mathcal{S}} \sqrt{n_i}W_i$.

Fluc computes the total number of queue buffers requested in a time period, Q_{curr} , and uses it to manage the buffer queue in three modes of operation, namely (a) uncongested mode, where $Q_{curr} \leq Q_{min}$; (b) congested mode, where $Q_{min} < Q_{curr} \leq Q_{max}$; and flooding mode, where $Q_{curr} > Q_{max}$. The activation of the token-bucket mechanism begins in the congested mode (b) with the initial parameters for a path identifier S_i set to T_{S_i} and $N_{S_i}^T$, respectively (viz., Eqs. (3.1) and (3.3)).

Uncongested Mode. If $Q_{curr} \leq Q_{min}$, all packets are serviced regardless of token availability. The router’s buffer queue tolerates temporary bursts of traffic until packet arrivals fill it. Link under-utilization, which may be caused by unnecessary

packet drops, is avoided. However, attack-path flows may still appear in this mode and consume more buffers (and higher bandwidth) than legitimate flows until the router queue reaches Q_{min} . In this case, FLoc forces entry in congested mode as soon as $Q_{curr} > Q_{min} \times \min\{1, \frac{C_{S_i}}{\lambda_{S_i}}\}$, where λ_{S_i} is the request rate of path S_i . This test leads to the activation of the token-bucket mechanism for attack path identifiers early, and causes them to experience packet drops before legitimate ones.

Congested Mode. If $Q_{min} < Q_{curr} \leq Q_{max}$, the token-bucket controls are activated for all path identifiers in the queue. However, since FLoc underestimates the token-buffer parameters (as discussed above), some path identifiers may experience unnecessary packet drops. To avoid penalizing legitimate path identifiers with unnecessary drops, FLoc implements a *random-drop* (i.e., neutral) policy in congested mode, instead of a targeted per-path drop policy, as required by the (under-estimated) token-buffer parameters. That is, if a packet does not get a token on its arrival, a queue threshold value, Q_{th} , is picked at random between Q_{min} and Q_{max} , and the packet is dropped only if $Q_{curr} > Q_{th}$.⁶ The random drops end when the *uncongested mode* is re-entered, namely when $Q_{curr} \leq Q_{min}$.

Flooding Mode. If $Q_{curr} > Q_{max}$, then either traffic bursts or unresponsive and/or attack flows persist. In either case, FLoc applies the packet-admission (drop) policy (viz., Eq. 4.5) defined by the token-bucket mechanism with the bucket size N_{S_i} instead of $N_{S_i}^T$.

Handling Attack Flows. To compute the flows' MTD, FLoc maintains a

⁶The random threshold (Q_{th}) functions as an early congestion notification much like the RED queue (the drop probability goes up as the queue length grows), yet it does not require complex parameter calibration as the RED queue does.

packet-drop history table (PDHT) whose entries comprise the flow identifier (f_i), $MTD(f_i)$, drop count ($D_{S_i}^{f_i}$), the last update time ($t_l(f_i)$) and the expiration time ($t_x(f_i)$) of an entry. FLoc implements the following simple mechanism for MTD accounting.

Suppose that a packet drop on f_i occurs at time t . If f_i already has an entry in PDHT, we update the entry as follows.

$$\begin{aligned}
 D_{S_i}^{f_i}(t_j) &+ = 1 \\
 MTD(f_i) &= \frac{1}{D_{S_i}^{f_i}}(t - t_l(f_i) + (D_{S_i}^{f_i} - 1) \cdot MTD(f_i)) \\
 t_x(f_i) &= t_x(f_i) + nMTD(\mathcal{F}_{S_i}) \\
 t_l(f_i) &= t
 \end{aligned}$$

If f_i does not exist in PDHT, a new entry for f_i is created, and $t_l(f_i)$ is set to t . Both $MTD(f_i)$ and $t_x(f_i)$ are initialized to the *reference* MTD, namely $nT_{S_i} = \frac{W_i}{2}RTT_i$. Since we only need to keep track of attack flows, a table entry whose $MTD(f_i)$ is greater than the reference MTD ($n_iT_{S_i}$) expires and can be replaced by a new flow. Once an attack takes place, the MTDs of both legitimate and attack flows drop below the reference MTD, yet an attack flow's MTD would be much lower than that of a legitimate flow.

Restricting the bandwidth of attack flows secures more bandwidth for the legitimate flows and hence leads to legitimate flows' MTD increases. Based on this observation, we apply the packet admission policy defined by Eq. (3.5) only to those flows that (1) have smaller MTD than the average MTD of a path, and (2) have experienced packet drops at least W times. As attack flows are penalized, a

legitimate flow’s MTD approaches, or exceeds, the reference MTD.

3.6 Simulation Results

In this section, we present our ns2 simulation results for various attack scenarios to evaluate our design. We use the tree topology shown in Figure 3.2, where both the height and degree of the tree are set to three (i.e., 27 paths). We attach 30 legitimate (TCP) sources to every leaf node, and attach 60 additional attack sources to each of 6 leaf nodes designated as attack nodes (i.e., we use 360 attack sources). Each legitimate source is configured to send a 12MB file to a destination server located across the link targeted for flooding and randomly starts its transmission between zero and five seconds. Each attack source is configured to change the *send* rate from one to ten times its fair bandwidth depending on the simulation scenario. The target-link capacity is set to 500 Mbps.

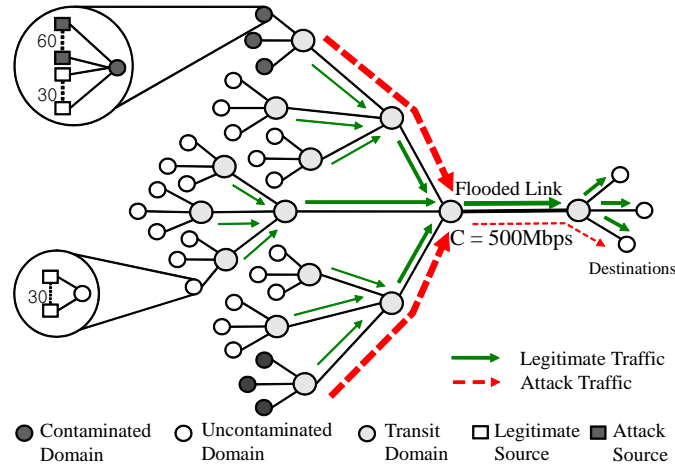


Figure 3.2: Simulation Topology

3.6.1 Attack Confinement

We illustrate the attack-confinement effects (e.g., bandwidth guarantees to legitimate path identifiers) during link flooding for three different types of attacks: a high-population TCP attack (Figure 3.3(a)), a Constant Bit Rate (CBR) attack (Figure 3.3(b)), and a Shrew attack (Figure 3.3(c)) [XG05]. The aim of the high-population TCP attack is to reduce the bandwidth of legitimate flows at a congested router. These flows adapt their *send* rate to the available bandwidth and as a consequence become indistinguishable from legitimate flows at that router. However, FLoc confines the effect of this attack to a single path identifier, since bandwidth is separately guaranteed to each path. As Floc guarantees the same bandwidth allocation to each of the 27 paths shown in Figure 3.2 (i.e., $500 \text{ Mbps} / 27 = 18.5 \text{ Mbps}$ per path), the bandwidths received by individual path identifiers shown in Figure 3.3(a) are almost identical regardless of their (legitimate or attack) population.

In the CBR attack, each of the 360 attack sources (i.e., “bots”) sends 2 Mbps CBR traffic through the targeted link. Thus the overall attack strength reaches 720 Mbps – an amount that would disrupt most legitimate TCP flows through a 500 Mbps link. Figure 3.3(b) shows that the legitimate-path flows get higher bandwidth in this attack than in the high-population TCP attack. This is because the token-bucket mechanism is activated early for attack paths and the fixed (non-increased) token-bucket sizes limit the traffic on these paths. At the same time, attack flows are easily identified by their low MTDs and are rate-limited accordingly.

In the Shrew attack, each attack source sends 2 Mbps traffic only during

0.25RTT seconds within an interval of RTT seconds. Also, we coordinate all attack sources to maximize the attack strength. Figure 3.3(c) shows that the bandwidths received by the flows of a legitimate path are almost identical to (or slightly higher than) those received by legitimate flows in the CBR attack. This means that the Shrew attack is handled at least as well as the CBR attack. Yet, the service rate has a higher variance during the Shrew attack. This is because flows that experience packet drops synchronized with the attack traffic utilize less bandwidth than the unsynchronized flows.

3.6.2 Robustness of Bandwidth Guarantees

We use the distribution of the bandwidth received by legitimate flows in legitimate paths to illustrate the robustness of FLoc’s bandwidth guarantees under various attack strengths. The strength of FLoc in this area is compared with that of an aggregate-based defense scheme (i.e., Pushback [MBF⁺02]) and a flow-based defense scheme (i.e., RED-PD [MFW01]). Figure 3.4 shows the cumulative distribution function (CDF) of the bandwidth received by the flows of legitimate paths (measured in a 20 to 80 second interval) for various attack strengths. (The *send* rate of each attack source specified in each graph’s legend is increased starting from 0.2 Mbps.)⁷ In Floc’s case (Figure 3.4(a)), the bandwidth distributions to flows of legitimate paths are nearly identical for various attack strengths, and the mean bandwidth is close to the ideal fair bandwidth; i.e., 18.5 Mbps for each of the 30 legitimate paths yields

⁷We illustrate the bandwidth distribution of all flows in legitimate paths since the link bandwidth is allocated in equal amounts to all 27 paths (i.e., 18.5 Mbps per path). Also, we increase the send rate of each attack source starting from 0.2 Mbps, since this is the fair per-flow bandwidth allocation in attack paths (i.e., 18.5 Mbps / (60 attack + 30 legitimate flows) = 0.205 Mbps / flow).

0.617 Mbps per legitimate flow. Also, FLoc provides per-flow fairness comparable to that of the RED queue in normal (no-attack) case (viz., Figure 3.4(c)). RED-PD outperforms Pushback slightly in low-rate attacks, namely when overall rates are less than 500 Mbps, yet both RED-PD and Pushback do not provide effective protection to legitimate flows; viz., Figures. 3.4(b),3.4(c) where their CDF curves move left and drop below the “no attack” curve.

3.6.3 Effects of Different Path Delays on Bandwidth Guarantees

Next, we perform a simulation about the effects of different path delays (i.e., RTT_i s) on the bandwidth guarantees. In principle, TCP flows having low RTT_i s would consume more bandwidth than those having a high RTT_i s. Yet, this RTT_i effect on the flow bandwidth is expected to be insignificant with our scheme since our token bucket mechanism is designed in consideration of flow RTT_i s. For this simulation, we change the link delays of attack paths by the amount of Δ Delay (e.g., -15ms, +35ms and +85ms as shown in Table 3.1) as compared with those of the legitimate paths, and observe how the different link delays⁸ affect the bandwidth guarantees. As Table 3.1 shows, with our scheme, low RTT_i attack sources do not make any distinguishable effect on the flows in the legitimate paths. Meanwhile, for the case Δ Delay = -15ms, the less bursty nature of low RTT_i flows (due to smaller TCP window sizes) allows more bandwidth to the legitimate flows in attack paths. With

⁸Different path-delay configuration can be considered as the attack domains at different locations in the Internet.

Pushback and RED-PD, close attack sources to the destination significantly affects the bandwidth of the legitimate flows in attack paths (20 – 30% throughput loss as compared with that under remote attacks).

Table 3.1: Flow bandwidth for different RTT_i s. \mathcal{F}_{LP}^g denotes the set of legitimate flows in Legitimate Paths and \mathcal{F}_{AP}^g denotes the set of legitimate flows in Attack Paths. Δ Delay is the attack paths’ link delays relative to those of the legitimate paths. The mean and standard deviation of bandwidth are given in Mbps.

| | Δ Delay | $f_i \in \mathcal{F}_{LP}^g$ | | $f_i \in \mathcal{F}_{AP}^g$ | | $f_i \in \mathcal{F}^a$ | |
|----------|----------------|------------------------------|-------|------------------------------|-------|-------------------------|-------|
| | | Mean | Std | Mean | Std | Mean | Std |
| FLoc | -15ms | 0.615 | 0.128 | 0.424 | 0.086 | 0.065 | 0.007 |
| | 35ms | 0.582 | 0.097 | 0.397 | 0.107 | 0.042 | 0.013 |
| | 85ms | 0.600 | 0.095 | 0.399 | 0.135 | 0.0291 | 0.006 |
| Pushback | -15ms | 0.216 | 0.057 | 0.009 | 0.007 | 0.932 | 0.446 |
| | 35ms | 0.303 | 0.071 | 0.010 | 0.008 | 0.786 | 0.452 |
| | 85ms | 0.311 | 0.066 | 0.008 | 0.006 | 0.767 | 0.435 |
| RED-PD | -15ms | 0.199 | 0.035 | 0.190 | 0.034 | 0.883 | 0.100 |
| | 35ms | 0.249 | 0.037 | 0.144 | 0.027 | 0.850 | 0.079 |
| | 85ms | 0.255 | 0.035 | 0.113 | 0.028 | 0.853 | 0.083 |

3.6.4 Differential Bandwidth Guarantees

Next, we evaluate the differential bandwidth guarantees achieved by FLoc’s path-aggregation policies. We set the maximum number of bandwidth-guaranteed paths to 25 (i.e., $|\mathcal{S}|_{max} = 25$ of the 27 paths) and allocate 20 Mbps bandwidth to each of them. This requires at least three out of six attack path identifiers of the contaminated domains in Figure 3.2 to be aggregated at the congested router. For legitimate-path aggregation, we place 15 legitimate-flow sources in each one of three sibling nodes (domains) and 30 legitimate sources in the other nodes. Since a third of 21 uncontaminated domains have 15 sources (i.e., 105 sources in all) and the others have 30 sources (i.e., 420 sources in all), there would be 525 flows originating

from legitimate domains at the congested router.

Figure 3.5 shows that without aggregation, nearly 80% of the legitimate-path flows receive less bandwidth than the other 20% of the flows (viz., the CDF marked by upward-pointing triangles). This implies that the flows of less populated paths (i.e., domains), which account for $105/525 = 20\%$ of all legitimate flows, consume much more bandwidth (i.e., two times more in our simulation) than those of highly populated paths (domains). This uneven bandwidth distribution disappears when legitimate-path aggregation is performed (viz., the CDF marked by circles).

Attack-path aggregation unavoidably penalizes legitimate flows of an attack path to some extent⁹ (but *never* denies link access to them). Since three attack paths are aggregated by the constraint of $|\mathcal{S}|_{max}=25$, the legitimate flows of the *aggregated* attack paths only get a third of the fair bandwidth allocated to each path identifier. As the figure shows, these flows, which account for half of all legitimate flows of attack paths, receive somewhat less bandwidth than the legitimates flows of *non-aggregated* attack paths, and certainly less bandwidth than the legitimate-path flows – the expected result of differential bandwidth guarantees.

Figure 3.6 illustrates a comparison between the differential bandwidth guarantees provided by FLoc, and the bandwidths provided by Pushback and RED-PD, at different attack rates.

With FLoc, the bandwidth received by the flows of legitimate paths is over 80% of the link bandwidth, which is nearly identical to the proportion of legitimate

⁹This is the case for *all* aggregate-based defenses, including those where *a priori* information regarding the legitimacy of a flow path is given, such as CDF-PSP [CLSS08].

paths (i.e., $21/25 = 0.84$). Recall that there are twice as many attack sources as legitimate sources in contaminated domains. Consequently, the total bandwidth used by attack flows is higher than that of legitimate flows in the same paths (i.e., attack paths) even though per-flow bandwidth is higher for legitimate flows (viz., FLoc under 0.2 and 0.4 Mbps attacks in Figure 3.6). As attack sources increase their *send* rates, the traffic from those sources are more aggressively rate-limited by FLoc’s preferential drop policy. This leaves more bandwidth for legitimate flows in attack paths, while the bandwidth of legitimate-path flows remains unaffected. With Pushback, the bandwidth of legitimate-path flows decreases until the attack traffic dominates the link bandwidth and the packet-drop rate triggers the activation of rate throttling (i.e., in a “bandwidth soaking” attack [XG05]). Once rate throttling is performed, the bandwidth of the legitimate-path flows increases (viz., the last four bars for Pushback in Figure 3.6). However, the bandwidth of the legitimate flows in attack paths decreases significantly, since Pushback does not implement any per-flow measure to counter attack traffic; i.e., “collateral damage” within attack aggregates is unavoidable. RED-PD limits the bandwidth of attack flows more than Pushback for low-rate attacks, and protects legitimate flows in attack paths for all attack strengths. However, RED-PD is less effective in protecting legitimate-path flows (whose bandwidth is shown as white bars in the figure) than Pushback, when the send rates of attack sources are very high (e.g., 3.2 and 4.0 Mbps). This is because RED-PD allocates the same bandwidth to flows (including extraordinarily high-rate, attack flows) regardless of their send rates. FLoc outperforms both Pushback and RED-PD, both in terms of the bandwidth guaranteed to legitimate traffic and

link-bandwidth utilization for all attacks rates.

3.6.5 Covert Attacks

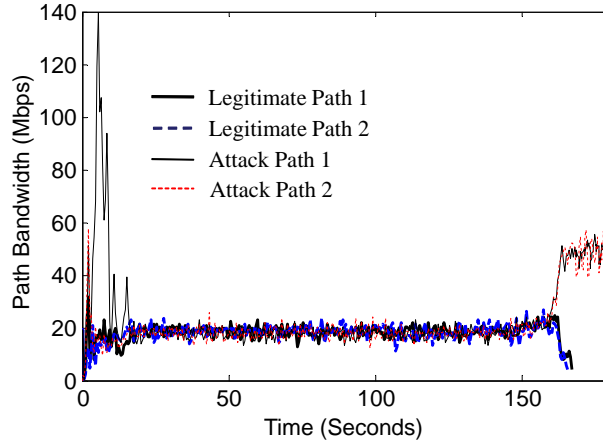
We evaluate the effect of covert attacks, where attack sources establish a large number of “legitimate-looking” flows through the congested link with “different IP destinations”, on FLoc, Pushback and RED-PD. For this simulation, we configure attack sources to connect with multiple destinations and to send low-rate CBR traffic (i.e., 0.2 Mbps per flow) to each destination to make individual attack flows look “legitimate”. (Recall that 0.2 Mbps is the fair bandwidth of each flow in attack paths; viz., Section 3.6.2.) The number of destinations to which an attack source connects concurrently within a router is increased from 1 to 20, which increases the send rate of individual attack sources from 0.2 Mbps to 4.0 Mbps. Note that since we use 360 attack sources in this simulation, the targeted link is already completely flooded at 7 connections per source (i.e., $360 \times 7 \times 0.2 \text{ Mbps} = 504 \text{ Mbps}$ which exceeds the link capacity of 500 Mbps). To illustrate the use of our covert attack countermeasures, we restrict the maximum number of concurrent connections per single source within a single router to 2 (i.e., two capabilities are made available to each multi-flow source, namely $n_{max} = 2$ ¹⁰), thereby limiting the bandwidth available to attack sources to 28.8% of the total link bandwidth. Of course, a source’s multiple connections through multiple routers are not affected by this restriction.

Figure 3.7 illustrates the bandwidth used by legitimate and attack flows at

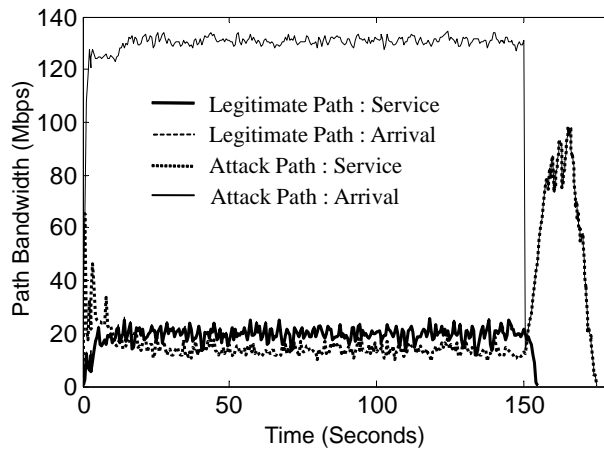
¹⁰We note that $n_{max} = 2$ is used only for the purposes of illustrating comparative performance analysis. We let n_{max} be a configurable parameter that can be differently chosen at different locations (i.e., routers).

the flooded link. Whenever an attack source increases the number of concurrent connections (i.e. flows) through a single target link, its legitimate-looking flows are classified as a single high-rate flow by FLoc. Hence, packets of the attack source are preferentially dropped at that router, much like those of CBR attack sources illustrated in the previous section. Pushback's reaction to these attacks is too late to make a difference. Its rate-control mechanism is triggered only at 12 flows per attack source when the maximum available link bandwidth is already exceeded by 52% (i.e., $360 \times 12 \times 0.2 \text{ Mbps} = 864 \text{ Mbps}$ vs. 500 Mbps maximum link bandwidth). Furthermore, Pushback neither prevents collateral damage within attack paths nor does it handle low-rate attacks (e.g., covert attacks whose total send rate is well below the maximum link bandwidth). RED-PD fails to counter covert attacks since bandwidth it provides to legitimate flows decreases as the number of attack flows increases. For example, when an attack source directs 16 concurrent connections through a single router, 810 legitimate and $16 \times 360 = 5760$ attack flows co-exist in that router. Hence, per-flow, fair bandwidth allocation provides more than 87.7% of the link bandwidth to attack flows.¹¹ This illustrates the lethality of covert attacks against typical schemes that act on either flow-aggregate or individual-flow basis to counter flooding attacks. Clearly, fair bandwidth allocation mechanisms cannot possibly counter such covert attacks.

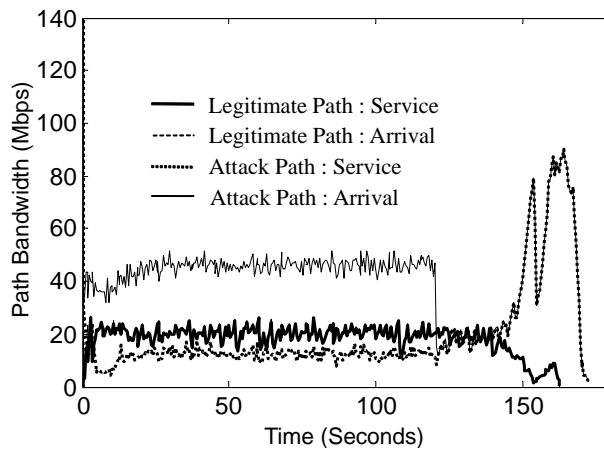
¹¹Legitimate TCP flows cannot fully utilize the allocated bandwidth due to their congestion control mechanism. This is why they use less than 10% of the link bandwidth in the simulation.



(a) High-population TCP attack



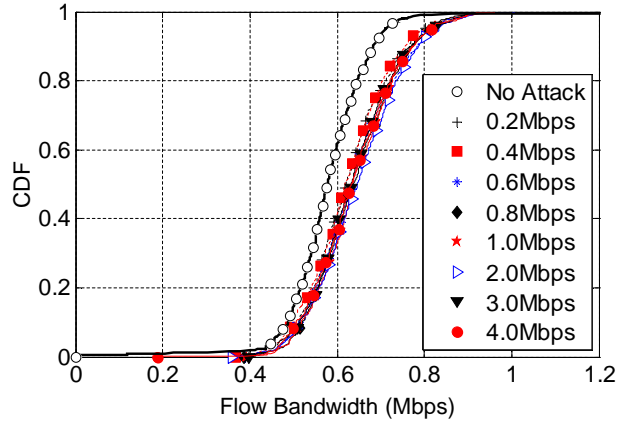
(b) CBR attack



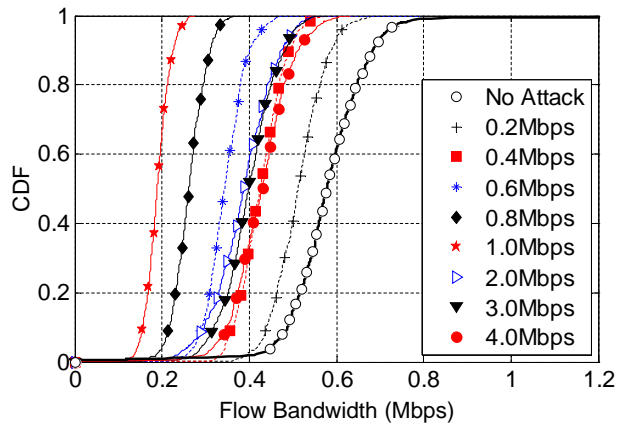
(c) Shrew attack

Figure 3.3: Localization effects for three different attacks. Legitimate and attack paths are randomly chosen from 27 paths.

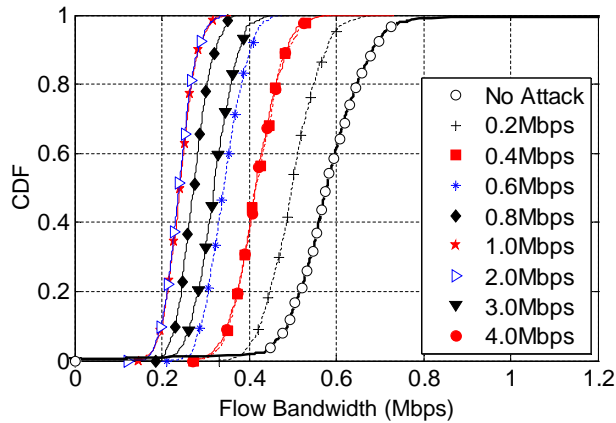
Legend: For each path identifier, “Service” denotes the bandwidth received, and “Arrival” the bandwidth requested



(a) FLoc



(b) Pushback



(c) RED-PD

Figure 3.4: Cumulative Distribution Function (CDF) of legitimate path flows' bandwidth for various CBR attack rates.

Note: For FLoc, the bandwidth of legitimate-path flows has nearly identical distribution independent of attack strength. In contrast, for Pushback and RED-PD, this bandwidth decreases significantly (i.e., the CDF curve moves left) as the attack strength increases.

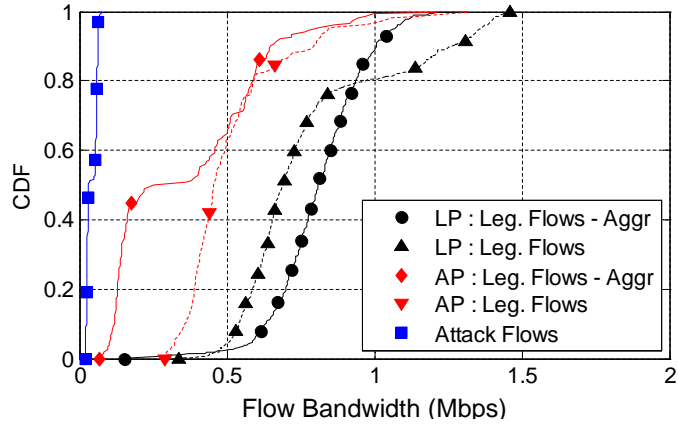


Figure 3.5: Differential Guarantees: Legitimate vs. Attack Flows. Legend: The CDFs labeled by “Aggr” illustrate the results of path-identifier aggregation. LP and AP denote Legitimate and Attack Paths, respectively.

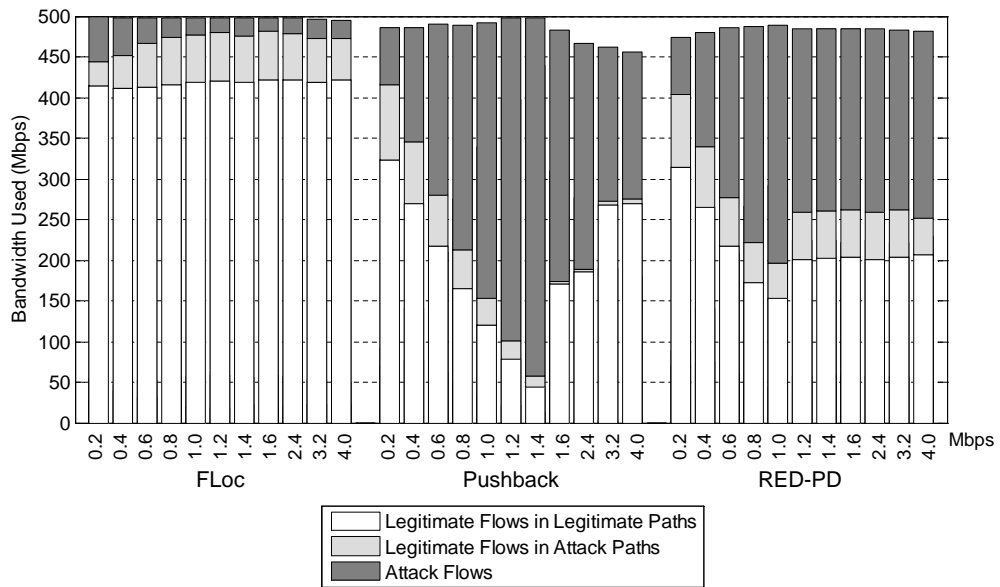


Figure 3.6: Differential bandwidth used at flooded link. (Flow rates of each attack source are shown on the horizontal axis.)

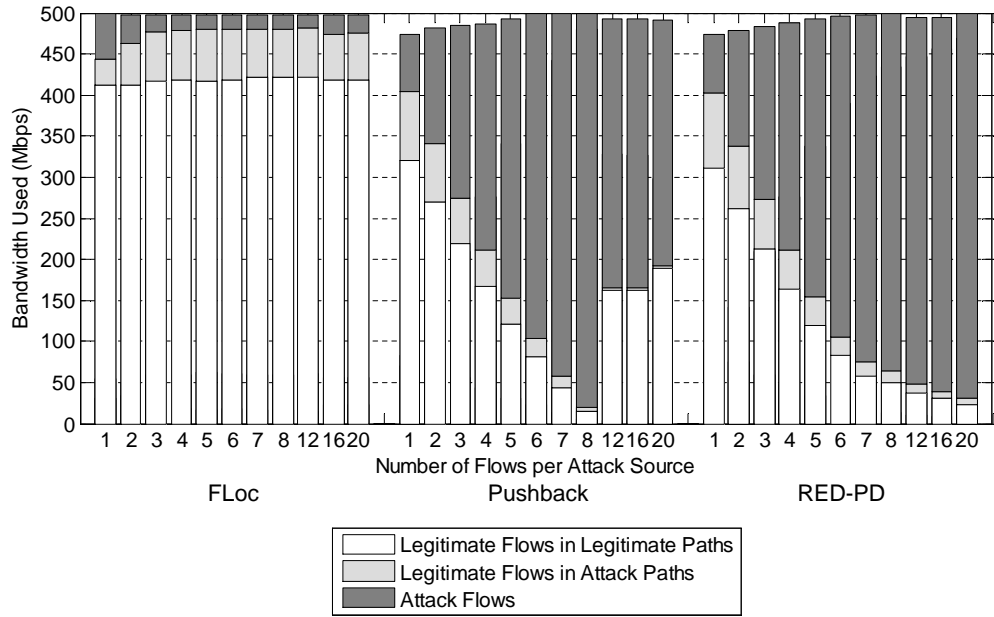


Figure 3.7: Bandwidth used at the flooded link under covert attacks.

Chapter 4

A Collaborative Defense against Link Flooding Attacks

4.1 Outline of the Chapter

In Section 4.2, we give a brief overview of related work. Then, we motivate our design and present an overview of our router-based defense scheme that consists of multipath routing, path-pinning and source-end rate control mechanisms in Section 4.3. We describe the implementation details of route control mechanisms and the rate allocation and control mechanism in Section 4.4. In Section 4.5, we simulate the potential side-effects of link-flooding attacks and provide simulation results showing that our scheme effectively mitigates those effects.

4.2 Related Work

Traffic Filters. Most of previous countermeasures against link flooding attacks work in a reactive way, which is seemingly reasonable since such flooding attacks occur rarer than usual. Generally, a reactive defense mechanism operates by identifying the flows or flow aggregates contributing to congestion and then installing filters or

rate-limiters on those flows. Attempts to handle attack traffic near its target have their strength in dealing with timed attacks (e.g., on-off and rolling attacks) and have no deployment obstacle since they can work independently. However, the side-effects of attacks on the way to the target (so called collateral damage to legitimate traffic [IB02]) cannot be handled by a sole, destination-end defense mechanism, and imprecise information on potential attack sources (due to spoofing or reflection attacks) limits their effectiveness. Installing filters close to the attack sources, either in a recursive way [IB02] or via a direct request to the domains that originate attack traffic [AC05a, SAM07, LYL08], would significantly mitigate collateral damages by preventing those attack traffic from being injected into the Internet. Yet, near-source filtering, despite its effectiveness, is less promising due to its lack of incentive to collaborating parties as pure source-end defenses [MPR02, GP01] are.

Address Authenticity. Recent capability-based approaches proactively prevent link flooding attacks by authorizing flows before they establish a connection. Network-layer capability, which is initially proposed in [ARW03] and extended in [YPS04, YWA05, YWA08], has great significance in preventing address spoofing and filtering out unwanted traffic at the packet destinations. Yet, another approach is to design an accountable Internet Protocol (e.g., [ABF⁺08]). However, in the absence of strong authentication mechanism at the network-layer (which is impractical at this layer), network links are still vulnerable to the flooding attacks from a large number of compromised machines that use valid (meaning accountable) source identifiers.

Multipath Routing. Different from conventional defensive countermeasures that secure legitimate flows by filtering attack traffic, a variety of route-based approaches aim to avoid congestion by forwarding legitimate flows through less congested paths [XR06, AC04, ABKM01, YW06]. These approaches, though they differ in implementation details, commonly utilize path multiplicity of the Internet to avoid congestion in a source (domain) controlled manner. However, the path diversity could also be exploited by attack flows in the absence of a precise flow discrimination mechanism, whereby legitimate and attack flows are distinguished and then differentially routed.

4.3 Design Overview

In this section, we describe the basic mechanisms of our router-based defense scheme against link flooding attacks. Our defense scheme consists of three independent yet complementary functions, which are multipath routing, path pinning and rate control.

4.3.1 Problem Statement

Our defense scheme against link-flooding attacks is intended to localize the effects of attacks within a specified locale (e.g., a domains or neighboring domains), and at the same time, mitigate their potential side-effects in two ways: multipath routing and source-end rate-control. In principle, attack localization can be achieved if a traffic source would embed the path identifier into the packets it forwards and the

congested router allocates its resources in a per path-identifier basis. However, its effectiveness would be limited if (1) legitimate domains are located near highly contaminated region, and/or (2) frequent route changes occur due to link state changes (e.g., link failure). In the former case, flows from legitimate domains could experience significant congestion before reaching at the destination where per-domain bandwidth control is performed. Whereas, in the latter case, both inter-domain and intra-domain route changes would allow attack traffic to affect other legitimate traffic irrelevant to attacks, widening attack effects. We mitigate those undesirable effects of flooding attacks by introducing a complementary routing policies (i.e., multipath routing and path pinning) as both problems are related to the routing policies in the Internet.

Even if new routing policies could localize/mitigate the attack effects to a significant extent, they are insufficient as the collateral damage to the legitimate flows sharing the same paths with those flows are unavoidable. Countering such collateral damage requires a source-end defense mechanism [SAM07,AC05a,MPR02]. Yet, deploying such countermeasures are less promising if no tangible/substantial benefit could be provided to the cooperating source domains. We argue that providing incentives to cooperating source domains is crucial for DoS defense, and hence design a rate-allocation and control mechanism favoring flows from such domains, i.e., a mechanism for rewarding policy-compliant domains.

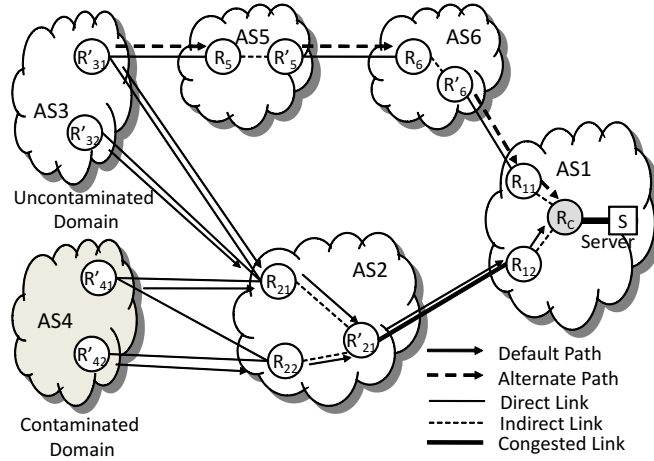
4.3.2 Assumptions

We assume that every packet carries the path-identifier described in Section 2.3.2 in its header. We use this path-identifier as a domain identifier mainly because it is secure (meaning unforgeable) and does not require universal deployment. We also assume that each domain has private-public key pair that is certified by a trusted third party (e.g., ICANN).

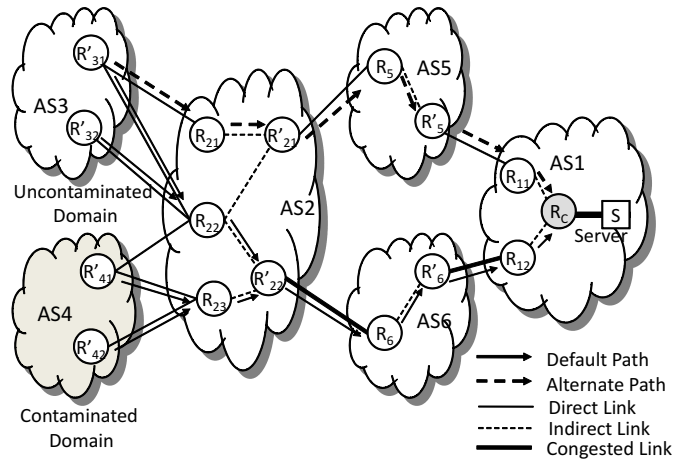
4.3.3 Multipath Routing

In rerouting legitimate flows, we utilize the multiplicity of inter-domain paths (AS paths) for at least a couple of reasons: (1) AS paths are spontaneously disclosed in the BGP protocol and hence no extra protocol or protocol change for path disclosure is required, and (2) the stability of AS paths (since they are determined by a path-vector protocol and the policies of individual domains) renders the effects of path change predictable, hence could make such path control effective.

AS path diversity is comprehensively explored in MIRO [XR06], which evidences that most of the domains (at least 95% of 300 million AS pairs tested) have alternate paths when 1-hop immediate neighbors' are counted (we will explain how those paths can be used below). Motivated by MIRO, we utilize the path multiplicity in an end-to-end (in terms of domain) controlled manner – we call it mutually controlled multipath routing – rather than a neighbor-negotiable way presented in MIRO. End-to-end path control has its strength in handling attacks since a traffic origin could recognize the occurrence of an attack (or persistent congestion) with-



(a) Multi-homed



(b) Single-homed

Figure 4.1: Multipath Routing : AS3 is an uncontaminated domain whose traffic is to be directed to an alternative path, and AS4 is a contaminated domain that sends attack traffic. (a) AS3 selects a different provider (AS5) in forwarding packets to the congested link, instead of AS2 (which has shorter AS path length to AS1). (b) The provider AS (AS2) forwards the packets received from AS3 to a different path (i.e., through AS5).

out actively probing the network conditions (e.g., delay, packet drops) for specific destination prefixes (which is nearly impractical in the absence of any signaling mechanism by destinations) and no protocol extension for on-line path negotiation is required to BGP routers.

The mutually controlled inter-domain routing works as follows.

Request from Destination. In order to reroute the traffic originating from an uncontaminated domain experiencing severe congestion, a congested (or attacked) domain issues a reroute request to the source domain. The reroute request includes the domains to be avoided in the forwarding path and the list of preferred domains (ordered by their priority) through which arriving traffic would experience less congestion.

Routing at Source. On receiving a reroute request, the source domain refers its routing table to find an alternate path to reach the destination via the requested route. If such path does not exist, the source AS selects another path that excludes the domains to be avoided. The alternate path found in BGP table is set to the default path for the destination prefix. The control message for multipath routing is exchanged between “route controllers” located in individual domains, where the route controller communicates with all BGP routers in its own domain to gather congestion information and apply route selection policies. We will describe the implementation details in section 4.4.1.

For a single-homed source domain (i.e., when no alternate path is available at the source domain), the congested domain requests rerouting to its provider that usually has multiple connections to tier-1 or tier-2 ASes, or is a tier-2 AS in itself. A provider AS, if it provides service to both contaminated and uncontaminated domains (as shown in Fig. 4.1(b)) should be able to forward only legitimate traffic (informed by the congested domain) to an alternate path, forwarding the traffic received from others (those domains could be classified as contaminated or have lower service priority at the destination) through the default path. Such differential

routing service can be provided to a customer AS by setting up a tunnel for a specific destination prefix to the designated next hop AS and forwarding others through the default path (e.g., AS2 forwards AS3's traffic to AS5, while leaving other's (AS4) to follow the default path via AS6).

4.3.4 Path Pinning

While utilizing path diversity for legitimate traffic, we need to prevent attack (or suspicious) traffic from being benefited by it. During flooding attacks, routes to the target link may exhibit the intrinsic path diversity of the Internet in several ways: a BGP session failure due to a severe congestion (or a flooding attack) causes a route withdrawal that makes the upstream AS choose an alternate path, link-state changes (e.g., link failure) within a domain could shift traffic to follow an alternate (intra-domain) path, and intra-domain path changes could propagate to other ASs. Such route changes are undesirable because route changes would shift attack traffic to other paths, making their effects reach at other legitimate flows. We counter those problems by pinning the routes of attack flows in the same controlled manner as the multipath routing presented in the previous section.

A congested domain sends path-pinning requests for high-rate attack flows to source/provider domains that originate/forward those flows (we will explain how to identify high-rate attack flows in Section 4.4.3.1). Whenever the route controller of a source domain receives a path-pinning request, it configures the BGP routers of the domain to suppress any route update message on the requested destination

prefixes, received from downstream domains. This leaves the default route to the destination prefixes unchanged. If a path-pinning request is made to a provider, the provider constructs a tunnel for the flows destined to the flooded link, which minimizes the collateral effects of those traffic.

4.3.5 Rate Control

The effects of flooding attacks can be mitigated by multipath routing and path pinning, yet those effects still persist unless attack flows are blocked near their origin. For example, legitimate flows originating from highly contaminated domains would share their fate with attack flows in the absence of precise flow distinction (between legitimate and attack flows) and corresponding authorization mechanism. Besides the collateral damage to the flows of the same domain of origin, attack flows would also affect other flows having different destination prefixes once they leave their origin [IB02]. Ideally, it is desired to block attack traffic near their origin as implemented in Pushback [IB02] and AITF [AC05a]. However, previous near source rate-control and filtering schemes are less attractive because they lack any positive incentive to cooperating source domains.

To encourage source-end treatment of attack traffic, we propose an end-to-end flow classification and throttling mechanism, where source domains determines the service priorities of flows guided by a congested domain; and, based on the priority, the congested router (in the congested domain) controls the flows' bandwidth. The basic mechanism works as follows. The route controller of a congested domain

requests a source/provider domain controller to classify flows into three categories – high priority, low priority, and to be filtered – by providing two threshold values (i.e., B_{min}^{th} and B_{max}^{th}). The egress router(s) of the source domain marks zero on the packets heading to the congested link at a rate of B_{min}^{th} (high priority); one at a cumulative rate of B_{max}^{th} (low priority); and two on the remaining packets. The congested router would offer guaranteed services to the high priority packets, a best effort service to low priority packets, and drop the other packets if congestion persists. The congested router implements separate queues for three different classes of flows, and places packets to the corresponding queues based on their marking. As such, the congested router provides additional bandwidth (though not guaranteed) to the packets of marking compliant domains that would otherwise be dropped.

We envision that the above rate-control scheme would provide sufficient deployment incentive to source domains for several reasons. For example, source/provider domains (1) can provide better service to selective customers, (2) do not need to strictly limit outgoing traffic rate on behalf of other domains, and (3) would be allocated more bandwidth at the flooded link by conforming to a marking request.

4.4 Architecture

In this section, we describe the network architecture and implementation details for performing the desired functions presented in the previous section.

4.4.1 Route Controller

Our route-based defense mechanism against flooding attacks introduces a specialized server, named “route controller”, in each domain to control traffic routes. Route controllers have different roles depending on its location. The route controller of a congested domain is responsible for handling congestion notification from its domain routers and sending route control message to source/provider domain controllers. A rerouting message is sent to uncontaminated domains, and a path-pinning and packet marking (classification) messages are sent to contaminated domains. The recipient of route control message identifies the BGP routers that can handle the message and configures the routers to direct flows to the flooded link as requested. Fig. 4.2 illustrates these route-control message flows. We describe the control message format in Section 4.4.4.

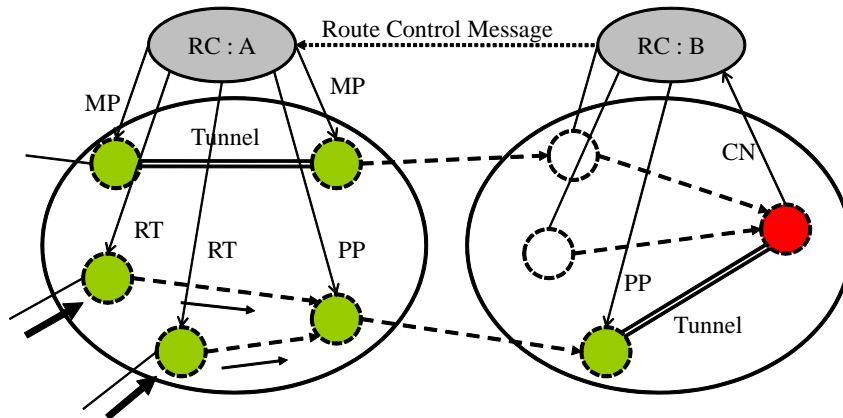


Figure 4.2: Route Control Messages: The congested router sends congestion notification (CN) message to its route controller (RC:B), and the message is forwarded to the route controller (RC:A) that can handle the requested message. The multipath routing (MP) message establishes a tunnel between ingress and egress routers; the path-pinning message (PP) suppresses route update message from the downstream domains; and the rate-throttling message (RT) makes responsible routers mark/drop packets.

To perform route-control operations, route controllers should (1) have the complete knowledge on the network topology of its domain and (2) be able to establish a session with BGP routers¹. Route controllers can satisfy the first requirement by either overhearing of or participating in intra-domain routing protocol (i.e., IGP). The second requirement can be easily configured since the route controller and BGP routers are under the same administrative control.

4.4.2 Route Management

During flooding attacks, a congested router constructs a traffic tree from the path-identifiers it has received, whereby source domains and their domain paths (AS paths) to the congested router are identified. For the path-identifiers collected, the congested router estimates the proportion of attack traffic that each path-identifier delivers (by measuring the send rate of each path-identifier), defined as the “domain contamination.” We will discuss how to estimate domain contamination in section 4.4.3.1. With the path-traffic information gathered as such, the congested router identifies two sets of domains: (1) highly contaminated domains that forward high volume of traffic, and (2) highly “influenced” domains that are not contaminated yet highly affected by attack traffic due to their locales (i.e., share much of their paths to the destination with attack traffic).

The flows from highly influenced domains are desired to be rerouted; and the flows from highly contaminated domains to be forwarded on the “pinned” default

¹A BGP router is generally configured to establish BGP sessions only with its neighbors to prevent false route advertisement.

paths. For rerouting, the congested router sends a reroute request to the route controller of its domain. A route request consists of the destination address prefix to the congested link and a list of related domains for rerouting, which are a source/provider (AS_S) that preforms rerouting, a preferred incoming domain(s) (AS_I^P) through which rerouted traffic is forwarded, and the congested incoming domains (AS_I^C) that need to be avoided. Then, the route request is delivered to the route controller of AS_S . A path-pinning request is sent in the same way, yet it is delivered to a provider domain that hosts the contaminated domains. The rerouting and path-pinning mechanisms at the source/provider domain is performed as follows.

4.4.2.1 Domain Discrimination

A congested router identifies contaminated domains that send high volume of attack traffic by the drop rate of path-identifiers, denoted by D_{S_i} (recall that the path-identifier is the ordered list of ASs through which a packet is forwarded). We define a path-identifier whose drop-rate exceeds a certain threshold D_{th} (e.g., $D_{th} = 10\%$ of allocated bandwidth C_{S_i}) as an attack path-identifier, and denote the set of attack path-identifiers by \mathcal{S}^A . Then, path-identifiers that (1) traverse the same AS(s) with attack path-identifiers and (2) send packets at a lower rate than the allocated bandwidth (C_{S_i}), could be highly affected by attacks and need to be rerouted (named attack-influenced path-identifier). These are formally defined as follows.

- \mathcal{S} : set of all path-identifiers.
- \mathcal{S}^A : set of attack path-identifiers,

$$\mathcal{S}^A = \{S_i | D_{S_i} > D_{th}\} \text{ for } S_i \in \mathcal{S}.$$
- \mathcal{S}^L : set of legitimate path-identifiers, $\mathcal{S}^L = \mathcal{S} - \mathcal{S}^A$
- \mathcal{S}^I : set of attack-influenced path-identifiers,

$$\mathcal{S}^I = \{S_i | \lambda_{S_i} < C_{S_i}, S_i \cap S_j \neq \emptyset\} \text{ for } S_i \in \mathcal{S}^L \text{ and } S_j \in \mathcal{S}^A.$$
- l_{S_i} : impact length of S_i ,

$$l_{S_i} = |S_i \cap S_j| \text{ for } S_i \in \mathcal{S}^L \text{ and } S_j \in \mathcal{S}^A.$$

In the above definition, the impact length of legitimate path-identifier S_i is the largest number of AS hops S_i shares with an attack path-identifier. This value will be used in selecting rerouting policies listed below.

4.4.2.2 Rerouting

A route controller, depending on its location (i.e., source or provider), performs rerouting in different ways.

Source Domain. A source domain can perform rerouting only when it has multiple AS paths to the requested destination (i.e., is multi-homed). When the domain controller of a multi-homed source domain receives a reroute message, it first selects the next-hop AS to which forwarded traffic would reach at the preferred incoming AS by the congested router. If multiple distinct next-hop ASs satisfying the reroute request (i.e., multiple AS paths) exist, the route controller selects an

AS based on the priority described in the route selection process [bgp]. The router controller sets the selected AS path as the default path for forwarding traffic to the congested link, by assigning the highest local preference value to the path. (“Local Preference” has the highest priority in the BGP route decision process.) The local preference value is set by the BGP router connected to the next-hop AS when importing route announcements from it; e.g., R'_{31} in Fig. 4.1(a).

Provider Domain. A reroute request at a provider domain is handled in the same way as the multi-homed source case if the provider is requested to reroute all its customers’ traffic. However, if the request is made for a specific set of customer domains, the provider would set up tunnels for the selected next-hop AS on behalf of those customers, leaving the default path intact, i.e., multipath routing. While tunneling can be implemented various ways (e.g., MPLS, IP-in-IP), we use an IP-in-IP tunneling mechanism in order to implement the tunneling protocol independently of, yet taking the advantage of a variety of intra-domain routing protocols for load balancing (multipath routing), service guarantees (e.g., RSVP), and performance improvement (e.g., MPLS). To do this, the route controller sends a tunneling request to the ingress router(s) (e.g., R_{21} in Fig. 4.1(b)) connected to the corresponding customer domain by providing the destination prefix that needs to be tunneled and the IP address of the egress router through which the next-hop AS is reached (Alternatively, the IP address of the next-hop AS’s ingress router can be used for tunneling.). The ingress router, on receiving the packet whose destination address matches the prefix requested for tunneling, encapsulate the original IP packet in the new IP packet destined to the egress router (e.g., R'_{21} in Fig. 4.1(b)). When the

egress router receives the packet, it decapsulates (i.e., peels off the outer IP header) the packet; and forwards the packet to the next-hop AS (we assume that each egress interface of the egress router has a distinct IP address.). We note that packet manipulation would not impose much overhead to BGP routers since it only applies to the flows carrying the requested destination prefixes. We also note that intra-domain traffic engineering, including potential intra-domain multipath routing (e.g., [KKDC05, MEFV08]), could be independently performed while network topology being kept private to the outside.

Destination Domain. A destination domain, if it has multiple incoming interfaces with upstream domains at different locations (i.e., border routers), can reroute legitimate traffic by changing the route export policies of border routers. Among multiple border routers in the next-hop domain, the upstream domain would select the router that announced the lowest MED (multiple exit discriminator) value, as the next-hop router. Hence, a destination domain can shift incoming traffic to a different path by changing MED values. This destination-based rerouting is most effective when the upstream domain is the traffic origin (i.e., impact length is 1).

A route controller selects one or combination of the above rerouting policies based on the AS connectivity and the impact length of a path-identifier; e.g., if the impact length of a path-identifier is greater than 2, rerouting at source or provider domain would be more effective.

4.4.2.3 Path Pinning

Path-pinning, which can be considered as a tunneling mechanism, aims to nail down the route of suspicious (attack) flows. When path-pinning is requested by the route controller of the congested domain, the recipient (the route controller of the source/provider domain) sets BGP routers to suppress the route update messages (received from neighboring domains) containing the requested destination prefix. Route update suppression would make the default path for the destination prefix unchanged, hence nail down the inter-domain path to the congested link. Meanwhile, any intra-domain route optimization (e.g., intra-domain multi-path routing) should not be performed on the flows carrying the requested destination prefix to confine their side-effects on other legitimate flows. This can be implemented using multi-topology routing [PMR⁺07, cis] – for which multiple routing topologies are kept in a router to forward traffic based on different criteria – by assigning a topology for this purpose.

As an alternative to the route suppression, a capability scheme can be applied as follows.

Capability Embedding. Implementing capability scheme [YPS04, YWA05] at the ingress router at which attack flows arrive would allow only authorized flows (by capabilities) to take advantage of any intra-domain routing optimization. In order to trade off the computational overhead of capability validation, we overload a flow’s routing information into the capability as follows.

An ingress router generates a capability for flow f_j by hashing the following

tuple with its secret key: (IP_S, IP_D, RID) , where IP_S is the source IP, IP_D is the destination IP, and RID is the egress router id. That is, f_j 's capability at router R_i is defined as $C_{R_i}(f_j) = RID || \text{Hash}(IP_S, IP_D, RID, K_{R_i})$. The packet destination returns this capability to the source for further packet transmission. Hence, address-spoofed packets or unwanted packets by their destination can be filtered at the capability-enabled router. We assume that a unique and private (i.e., meaningful within the domain) RID can be assigned to BGP routers of a domain and each RID can be mapped to the IP address of the corresponding router. Once a packet passes capability validation at the ingress router (i.e., authorized), it would be directed to the egress router distinguished by the RID in the capability.

4.4.3 Rate Throttling

4.4.3.1 Rate Allocation

As our defense mechanism is intended to favor flows from uncontaminated domains, distinguishing such domains from contaminated ones is essential. Of course, a congested router is able to allocate more resource (e.g., bandwidth, buffer space) for the traffic from preferred domains, yet difficulty in resource allocation arises if attack traffic is destined for public services and hence no preference regarding the traffic origins can be made. While a fair bandwidth allocation to all active domains would be a viable approach, it is less effective if attack sources are distributed in a large number of domains.

Estimating the proportion of attack flows originating from a domain (defined

as the domain contamination) enables a congested router to differentially allocate link bandwidth according to the domain contamination. Generally, domain contamination could be inferred by the aggregate transmission rate and/or a test on the “protocol compliance” (e.g., TCP congestion control, completion of TCP’s three-way handshake) of the flows originating from a domain. Additionally, we engage a domain’s compliance with the packet marking request – denoted by “policy compliance” – in router’s bandwidth allocation to reward policy compliant domains. This reward policy is both desirable and effective: it is desirable because reward would encourage source-end defense, which mitigates collateral damage to legitimate traffic; and it is effective because source domains can identify attack sources more precisely.

Let S_i be the path-identifier that represents a source domain, and \mathcal{S} be the set of all active path-identifiers seen at the congested router. And, let λ_{S_i} be the send rate of packets carrying S_i and C be the capacity of the congested link. Given λ_{S_i} for $S_i \in \mathcal{S}$, bandwidth allocation to S_i , denoted by C_{S_i} , is made as follows.

$$C_{S_i} = \frac{C}{|\mathcal{S}|} + \frac{C - \sum_{S_i \in \mathcal{S}} C \rho_{S_i}}{|\mathcal{S}^H|} \mathcal{P}_{S_i} \quad (\text{A-1})$$

where $\rho_{S_i} = \min\{\frac{\lambda_{S_i}}{C_{S_i}}, 1\}$, $|\mathcal{S}^H|$ is the number of over-subscribing domains (i.e., $\mathcal{S}^H = \{S_i | \lambda_{S_i} > \frac{C}{|\mathcal{S}|}, S_i \in \mathcal{S}\}$), and $\mathcal{P}_{S_i} = \min\{\frac{C_{S_i}}{\lambda_{S_i}}, 1\}$.

The first term in Eq. (A-1) is the guaranteed bandwidth to S_i (denoted by $C_{S_i}^0$), and the second term is the rewarded bandwidth to S_i on its policy compliance (denoted by $C_{S_i}^1$). The guaranteed bandwidth to S_i is determined as such (i.e., $\frac{C}{|\mathcal{S}|}$), in order to allocate the same (guaranteed) bandwidth to all domains. Whenever

the guaranteed bandwidths are not fully subscribed (e.g., some domains send less traffic than the guaranteed amount), the router has residual capacity that could be reallocated. This residual capacity is $C - \sum_{S_i \in \mathcal{S}} \frac{C}{|\mathcal{S}|} \rho_{S_i}$ and is reallocated to S_i proportional to its rate-control compliance \mathcal{P}_{S_i} – differential bandwidth reward. Thus, source domains that limit send rate to the congested link could be allocated additional bandwidth that would be unavailable otherwise.

4.4.3.2 Packet Marking

The congested router sends a packet-marking request (which implicitly requires rate-throttling) to the source/provider domains whose transmission rate exceeds the allocated bandwidth (i.e., $\lambda_{S_i} > C_{S_i}$). The packet-marking request includes two threshold values: the guaranteed bandwidth ($B_{min}^{th} = \frac{C}{|\mathcal{S}|}$) and the overall assigned bandwidth ($B_{max}^{th} = C_{S_i}$). On the request, the egress router of the source domain (or ingress router of the provider domain) writes high priority markings (i.e., 0) on the packets at a rate of B_{min}^{th} and low priority markings (i.e., 1) at a rate of $B_{max}^{th} - B_{min}^{th}$. And, it would either drop remaining non-markable packets to comply the rate-control policy of the destination or write lowest priority markings (i.e., 2) on them.

4.4.3.3 Rate Control

A congested router implements separate token buckets to provide bandwidth guarantees to path-identifiers. Each token-bucket (for a domain) consists of two sub-

buckets: a high-priority token bucket (denoted by HT_{S_i}) for the guaranteed bandwidth (i.e., $C_{S_i}^0$) and a low-priority token bucket (denoted by LT_{S_i}) for the rewarded bandwidth (i.e., $C_{S_i}^1$). The router controls the bandwidth of each path-identifier (domain) by applying the following packet admission policy. Let $\mathcal{S}^{\mathcal{L}}$ and $\mathcal{S}^{\mathcal{A}}$ be the set of legitimate and attack path-identifiers respectively. And, let $Q(t)$ be the current length of the high-priority queue and its (desired) normal operating range be $[Q_{min}, Q_{max}]^2$.

A packet is placed in the high priority queue if its path-identifier belong to

1. *Legitimate Path* and

- a token is available in HT_{S_i} , or
- a token is available in LT_{S_i} and $Q(t) \leq Q_{max}$, or
- $Q(t) \leq Q_{min}$.

2. *Priority-Marking Attack Path* and

- marking is 0 and a token is available in HT_{S_i} , or
- marking is 1 and a token is available in LT_{S_i} and $Q(t) \leq Q_{max}$.

3. *Non-Marking Attack Path* and

- high-priority token is available.

Thus, HT_{S_i} provides bandwidth guarantees to domains, and LT_{S_i} allows the residual bandwidth to be utilized by legitimate and priority-marking domains. Yet,

² Q_{max} is chosen to limit the maximum queueing delay and Q_{min} is chosen to allow instantaneous traffic burst. These values are assumed to be set by the router.

the router grants the tokens for the residual bandwidth (LT_{S_i}) only when $Q(t)$ stays in a normal operating range (i.e., $Q(t) \leq Q_{max}$), in order to handle instantaneous or potential traffic increase by the under-subscribing domains (which are allocated the same guaranteed bandwidth, namely $\frac{C}{|S|}$). Whenever $Q(t)$ goes below the minimum operating range (i.e., $Q(t) \leq Q_{min}$), the router enqueues the packets of legitimate paths regardless of token availability, to avoid link under-utilization during the flooding attacks (we set $Q_{min} = 20\%$ of the queue size in our simulation). This bandwidth control mechanism is illustrated in Fig. 4.3.

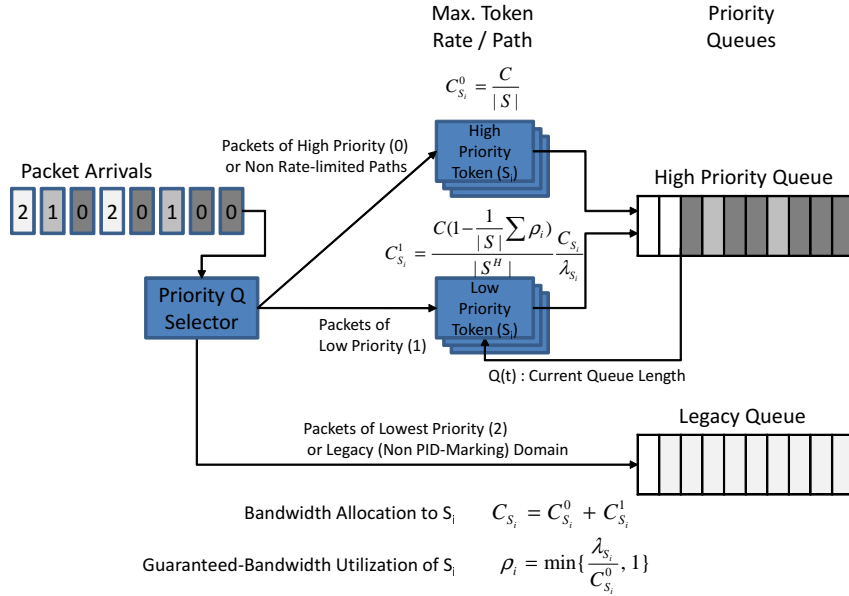


Figure 4.3: Rate Limiting at a Congested Router.

Packets that have lowest priority marking (2) are placed in the legacy queue (for non-prioritized, legacy³ traffic), which would be serviced only when the high-priority queue is empty. If the drop rate of a path-identifier exceeds a certain threshold (e.g., 10% of allocated bandwidth), it would be classified as an attack

³The packets that originate from non-path marking domains are denoted by “Legacy” packets and have low service priority at the congested domain.

path-identifier, and be requested rate-throttling.

4.4.4 Control Message Format

In this subsection, we describe the route-control message in detail. We show the message format in Fig. 4.4 and explain each field of the message below.

| | | | | | | | | |
|-----------------------|-----------------------|---------------------|-----------------|----------------------|----------------------|-----------|-----------------|-------------------|
| AS_s | AS_D | Addr. Prefix | Msg Type | Control Msg 1 | Control Msg 2 | TS | Duration | Sign of CD |
|-----------------------|-----------------------|---------------------|-----------------|----------------------|----------------------|-----------|-----------------|-------------------|

AS_s: Source Domain, AS_D: Congested Domain, TS: Timestamp

Figure 4.4: Control Message Format.

- *AS_S*: Source Domain of the flows that need to be controlled.
- *AS_D*: Congested Domain. The congested router sets this field to its id (which is uniquely assigned within the domain) when sending a congestion notification message to the route controller. Then, the route controller replaces the id with its own AS number and send the message to the route controller of SD.
- Addr. Prefix: Destination address prefix(es) of the flows that contribute congestion.
- Msg Type: Control message type. Each of Multipath routing (MP), Path-pinning (PP), Rate-Throttling (RT), and Revocation (REV) messages are assigned one bit from the lowest bit.
- Control Msg 1 and 2: These fields have different meanings depending on the message type. The message types and corresponding meanings (of Msg 1 and

Msg 2 separated by comma) are as follows.

MP - domains through which packets are routed (AS_I^P), domains to be avoided (AS_I^C).

PP - current AS path, unused

RT - minimum threshold (i.e., B_{min}^{th}), maximum threshold (i.e., B_{max}^{th}); viz., Section 4.4.3.2.

- TS: message creation time. TS is used to prevent replay attacks.
- Duration: duration of the route control. TS + Duration is the expiration time of the route-control request.
- Sign of CD: Digital signature of the control message signed by the congestion domain.

In the above message format, SD, Addr. Prefix, and Control Msgs fields can have multiple entries, hence the first byte of those fields used to indicate the number of entries. Inter-domain route-control messages (i.e., those between route controllers) are protected by a digital signature scheme (e.g., RSA, ECC signatures) for the authenticity and integrity of the message. And, intra-domain messages are protected by the message authentication code (MAC) that is generated using a pre-shared secret key between the route controller and each router in the same domain. We assume that every route controller has a private/public key pair certified by a trusted third party (e.g., ICANN), and shares secret keys with individual routers of its domain. When a route controller receives a route control message from the

congested router, the route controller verifies the MAC of the request and replaces the MAC with its signature if the MAC is correct.

4.5 Simulation

In this section, we evaluate the effectiveness of our scheme under various simulation scenarios using ns2. For simulations, we configure a network topology as illustrated in Fig. 4.5 and set various types of traffic (e.g., FTP, CBR, and Web traffic) to go through backbone links (i.e., background traffic)) to approximate real network condition.

Topology. Source domains (S1, ..., S6) and a destination domain (D) are attached to the provider domains (P1, P2, P3) and intermediate domains on the path (R1, ..., R7) connect them with two disjoint paths: P1 to P3 and P2 to P3. S1, S2, and S3 are attached to P1 and S4, S5 S6 are attached to P2. We set the bottleneck link capacity (i.e., the link between P3 and D) to 100 Mbps and the capacities of remaining links to 500 Mbps. In the topology, S3 has multiple providers (i.e., P1, P2) that have disjoint paths to the destination domain. The path between P1 and P3 is connected by three routers (each router represents an AS in our simulation) and the path between P2 and P3 is connected by four routers with higher link delays. Hence, S3 chooses P1 as the default next hop AS to the destination (we set this in our simulation by assigning lower link costs on S3's path to D through P1 than those on its path through P2.).

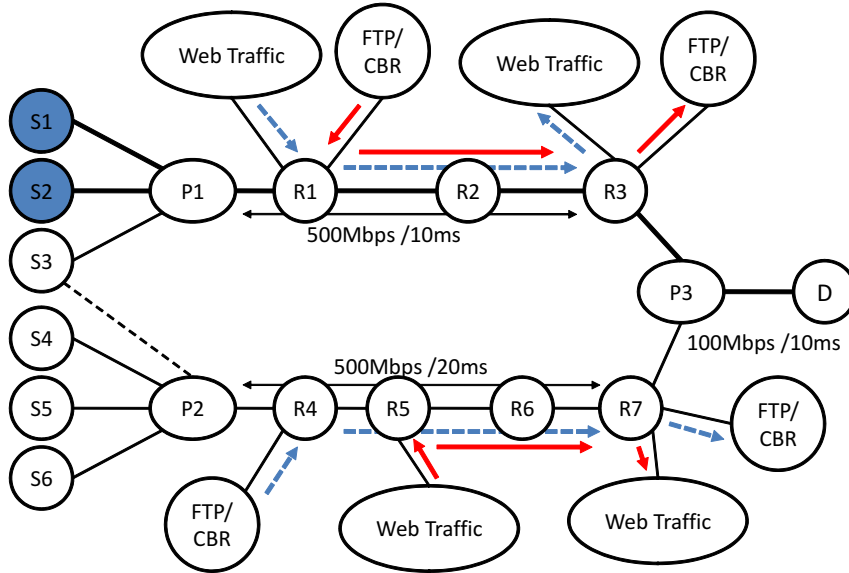


Figure 4.5: Simulation Topology

Traffic. To approximate real network-traffic conditions, we set Web (packet arrivals with Pareto distribution), FTP, and CBR (Constant Bit Rate) traffic to pass through the core network (denoted by R_i 's), which also delivers the flows whose bandwidth we want to observe (i.e., flows between S_i 's and D for $1 \leq i \leq 6$). We generate 300 Mbps Web traffic, 50 Mbps CBR traffic, and FTP flows originating from three different domains (i.e., three distinct paths seen at R_1 and R_4) for background traffic. We attach 30 FTP sources to each of source domains for generating legitimate flows, each of which sends 5 MB files to the destination D . Then, we measure the flows' bandwidth at the (attack) target link. As TCP flows adapt their send rate to available bandwidth and hence are highly vulnerable to link flooding attacks, their bandwidth at the congested link would reflect the worst effect of flooding attacks.

4.5.1 Multipath Routing

We place attack sources at two domains, namely S1 and S2, and configure the attack sources of each domain to collectively send 200 Mbps Web traffic (which can exhaust the bandwidth of the target link (100 Mbps) and the available bandwidth on the path to congestion controlling flows (i.e., 150 Mbps)). And, we set S5 and S6 send 10 Mbps traffic to observe how the under-subscribed bandwidth by those domains is re-allocated to other domains. The congested router (i.e., D) is enabled to perform per-path fair bandwidth control using a token-bucket mechanism [Par92] (where individual paths are assigned separate token buckets), and remaining routers implement drop-tail queues to model the legacy part of the network. To show the advantage of differential bandwidth allocation to the rate-limiting source domain, S2 is enabled to control the rate of outgoing traffic as requested by the congested router, while S1 forwards all packets destined to D to P1.

Fig. 4.6(a) shows the result of the scenario where S3 forwards traffic to the default next hop AS (i.e., P1). While the per-path bandwidth control at router P3 limits the bandwidth of S1 (i.e., attack domain) effectively, the high-rate traffic of S1 significantly affects the TCP flows originating from S3. Meanwhile, rate-controlling domain S2 uses more bandwidth than S1 (though less than that of legitimate domain S4) since the congested router D allocates the guaranteed amount of bandwidth to S1 and S2 (which is $100 \text{ Mbps} / 6 \text{ paths} = \approx 16.7 \text{ Mbps}$) yet allocates the part of the under-subscribed bandwidth by S5 and S6 (which is $33.4 \text{ Mbps} - 20 \text{ Mbps} = 13.4 \text{ Mbps}$) and is reallocated to S2, S3 and S4) to S2. As illustrated in the figure,

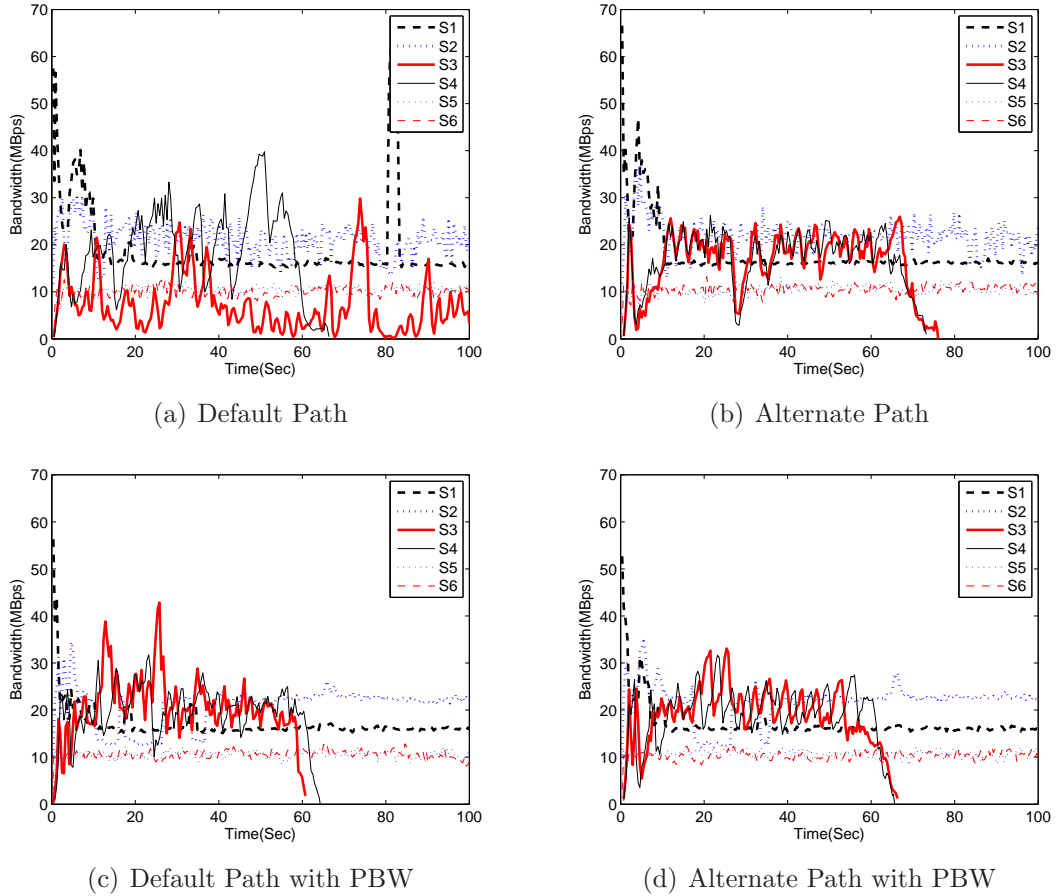


Figure 4.6: Bandwidth used by individual source domains at the congested link when the send rate of the attack domain (i.e., $S1$) is 200 Mbps. PBW stands for global deployment of path-bandwidth control at routers.

the Web traffic (which generally consists of short-lived TCP flows, e.g., flows from $S2$) is less affected by the flooding attack than long TCP flows (i.e., FTP flows of $S1$).

To protect the flows of $S3$, we employ multi-path routing at $S3$, thereby $S3$ routes its traffic heading to D via $P2$. Fig. 4.6(b) shows that the bandwidth of $S3$ is similar to that of $S4$ and bandwidth of all domains are stable when compared with the result of the single-path scenario: the variation of the bandwidth of $S4$ becomes lower since the bandwidth allocation at the congested router is stable (Router D

reallocates the under-utilized bandwidth by S1 to S4 in the previous simulation.).

To evaluate the effectiveness of multipath routing, we compare the result of multipath routing with that of global deployment of per-path bandwidth control at all routers (i.e., R1 to R7). As illustrated in Fig. 4.6(c), global per-path (fair) bandwidth control allows more bandwidth to legitimate domains (S3, S4) since it limits (1) the bandwidth of attack traffic near its origin and (2) the bandwidth of background traffic which could increase instantaneously (e.g., at 27 second in Fig. 4.6(b)). When both multipath routing and per-path bandwidth control are employed (viz., Fig. 4.6(d), the bandwidths of legitimate domains become more stable (i.e., less fluctuate over time) as they are less affected by the background traffic. Though effective, per-path bandwidth control cannot be easily deployed globally (considering the number of routers in the Internet) and would be less practical at the Internet backbone where a large number of flow-paths exist. Note that our multipath routing mechanism can be implemented at the Internet edges (i.e., stub/provider domains), hence does not require global change of routers.

Fig. 4.7(a) shows the simulation result of more aggressive attacks, where the aggregate send rate of attack domain is increased by 50% (i.e., 300 Mbps per attack domain). Increasing attack strength reduces the bandwidth of S1 more significantly in the absence of multipath routing or global per-path bandwidth control. Like the previous result, multipath routing protects the flows of S3 from flooding (viz., Fig. 4.7(b)). Yet, the bandwidths of S3 and S4 fluctuate due to the background traffic on the path between R4 and R7. This effect disappears when per-path bandwidth control is employed (viz., Fig. 4.7(d)). Fig. 4.8 compares the proportion of bandwidth

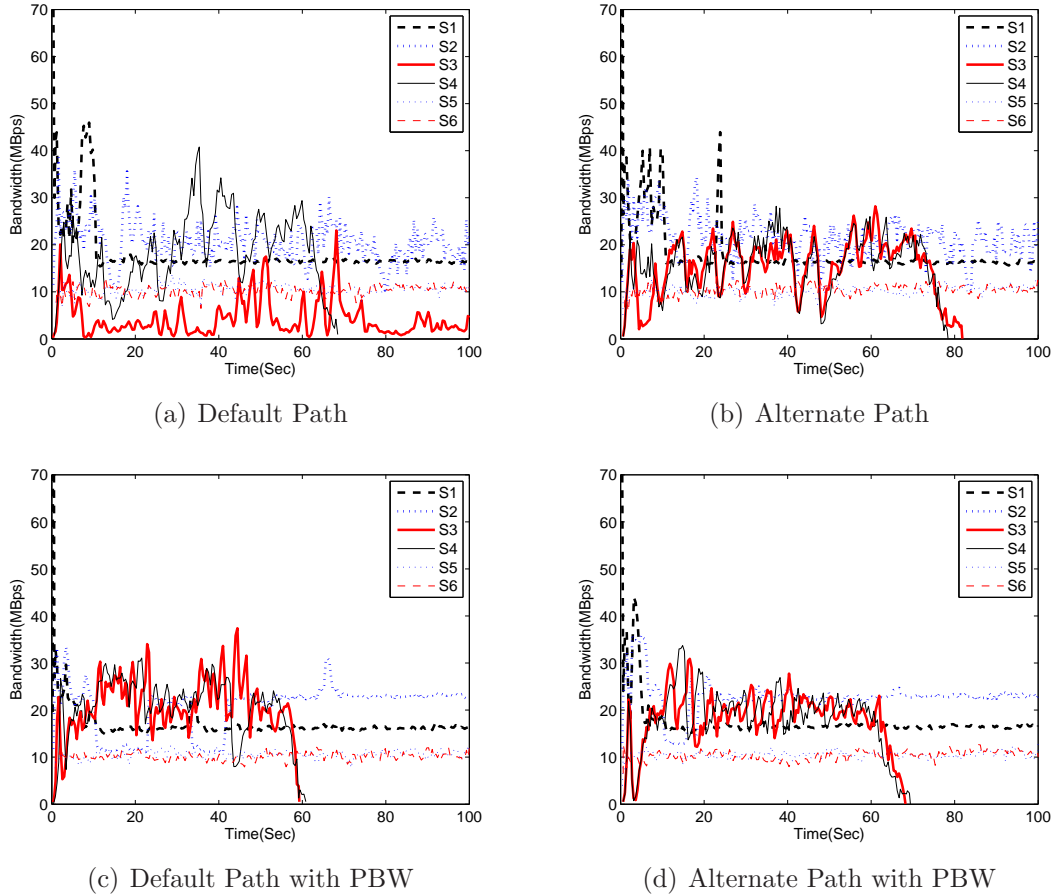


Figure 4.7: Bandwidth used by individual source domains at the congested link when the send rate of the attack domain is 300 Mbps.

used by each domain for different attack rates and defense strategies.

4.5.2 Effects of Attack-Path Change

We illustrate the potential effects of attack-route changes on legitimate traffic when the intermediate routers on forwarding paths do not perform per-path bandwidth control (if they do, attack effects would be limited as illustrated in the previous section.). For this simulation, we place attack sources at S3 (which is multi-homed as shown in Fig. 4.5, and change the next-hop AS of S3 from P1 to P2 at $t =$

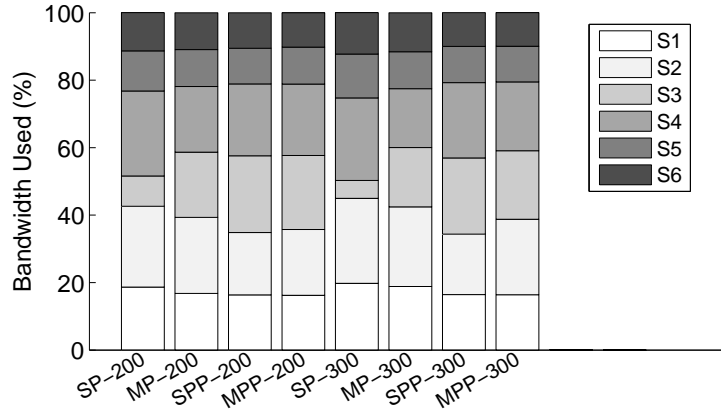
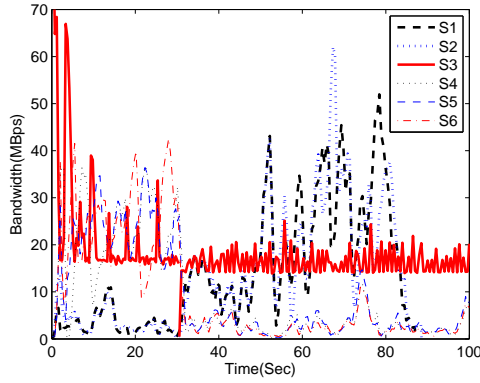


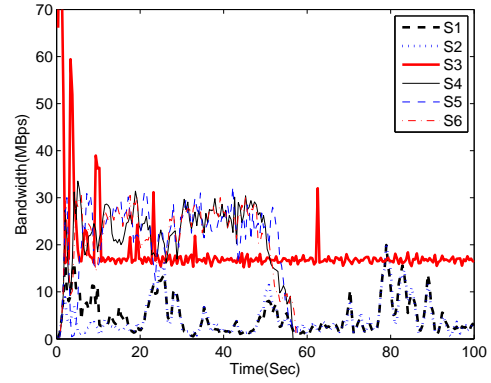
Figure 4.8: Bandwidth used by source domains at the congested link.
 Legend: SP: Single-path Routing, MP: Multi-path Routing, SPP: SP with global per-path bandwidth control, MPP: MP with global per-path bandwidth control. The numbers following dashes represent the send rate of each attack domain.

30 seconds. We simulate this path change in ns2 by employing the DV (Distance Vector) routing protocol and increasing the link cost between S3 and P1. We place FTP sources in the other domains and observe their bandwidth at the congested link. We run simulations for the attack strengths of 400 Mbps and 500 Mbps, and compare the results with those of the simulations where (attack) path-pinning is enabled.

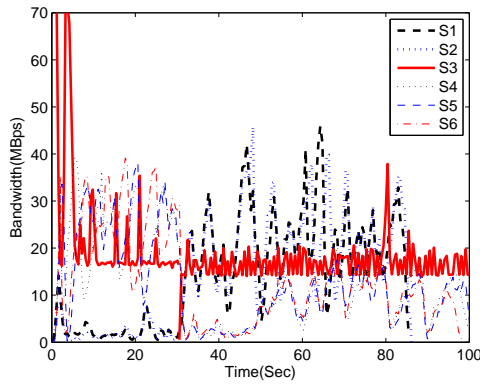
Fig. 4.9(a) and 4.9(c) show that the bandwidths of S4, S5 and S6 decrease significantly after attack route change occurs at $t = 30$ seconds even though the bandwidth of attack traffic is limited at the congested link. Route changes of attack traffic not only affect the flows that head to the target link but could possibly affect more legitimate flows (that have different destinations from attack traffic) since the alternate path chosen by a domain would be longer than the default path (recall that the path that has the shortest AS hops to the destination is chosen as a default path in BGP protocol). That is, attack effects are not confined within a specified



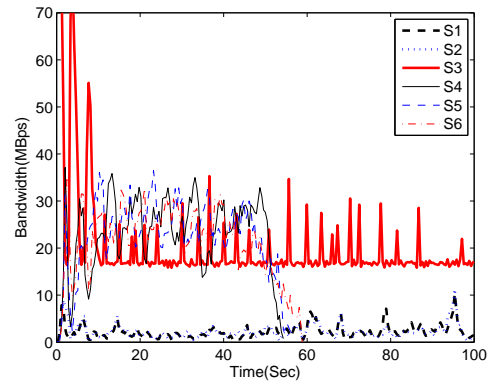
(a) Route Change under a 400 Mbps attack



(b) Path Pinning under a 400 Mbps attack



(c) Route Change under a 500 Mbps attack



(d) Path Pinning under a 500 Mbps attack

Figure 4.9: Comparison of bandwidths used by individual source domains at the congested link for different attack send rates (i.e., 400 Mbps and 500 Mbps) and different routing policies (i.e., single-path and multi-path routing).

locale (neighboring domains) but are dispersed widely. In contrast, when the route of S3 is pinned, the bandwidth of S4, S5 and S6 is stable as shown in Fig. 4.9(b) and 4.9(d).

Chapter 5

Conclusion and Future Work

In this chapter, we conclude this dissertation and discuss directions for future research.

5.1 Conclusion

In this dissertation, we have presented a comprehensive scheme that localizes the effects of link-flooding attacks with various respects and hence provides dependable link access (i.e., guarantees of link access) during the flooding attacks. We summarize the constituting mechanisms and reiterate our contributions below.

- *First*, we presented a new path identification mechanism that provides unforgeable domain identifiers to individual packets. Path-identifiers carried in packets enable remote routers to identify packets' domain of origin and help localize the effects of flooding attacks by employing per-domain based countermeasures. Our path identification mechanism is designed to be independently and incrementally deployed at the Internet edges (e.g., stub ASs) and its source-constructible nature eliminates the global deployment issue of previous packet marking schemes.

- *Second*, we presented a dynamic virtual queueing and path aggregation mechanisms that provide guarantees for network link access to capability request packets. These guarantees are provided in a domain basis, hence DoC attacks from contaminated domains, even if they are launched by a large number of bots, would not affect the link-access guarantees of legitimate capability requests that originate from uncontaminated domains. Simulation results in Section 2.7 show the strength of our scheme: precise guarantees independent of attack size and location, differential guarantees based on domain contamination, and resilience against timed attacks (e.g., Rolling attack).
- *Third*, we presented a per-domain bandwidth control mechanism called “FLoc” that provides precise bandwidth guarantees to the aggregate-flows of the same origin. In addition to the bandwidth guarantees, our mechanism can identify and rate-limit a variety of attack flows: flows of non-responsive/high-rate attacks, low-rate Shrew attacks, and more sophisticated “covert attacks” described in Section 3.4.2.3. Comprehensive simulations under those attack scenarios and comparisons of the results with those of other mechanisms show the effectiveness and robustness of FLoc especially for defending against the low-rate and covert attacks.
- *Finally*, we presented a complementary routing system that reroutes legitimate traffic via less congested paths using the path multiplicity of the Internet. Meanwhile, pinning the paths that deliver attack traffic prevents attack dispersion, hence localizes their effects. We implemented those functions by slightly

modifying the current routing policies at network edges (i.e., BGP routers), in order to make them easily adopted. In addition to the routing policies, the bandwidth reward policy applied at the congested router provides incentive for source-end defense that is essential for reducing the undesirable side-effects of flooding attacks (e.g., multiple congested links). Our simulation results illustrate the effects of flooding attacks on legitimate flows and how those effects are mitigated by our routing policies.

5.2 Future Work

Implementation. In this dissertation, we evaluated the effectiveness of our scheme by ns2 simulations. In addition, we plan to implement our scheme in a real system and evaluate its performance with real Internet traffic traces. This would provide more solid confidence on our scheme and its deployment benefit.

Economic Cost of Defense. Obstacles for deploying a new scheme range from the complexity of the scheme (i.e., in terms of resource requirement) to its economic cost. We will make a quantitative analysis on the economic cost of security-enhanced (meaning DoS-resilient) router design and deployment, and their cost-effectiveness. This also help identify essential functionalities that a new scheme needs to be equipped and places where the scheme should be deployed.

Intra-domain Extension. Our design mostly focuses on identifying and rate-limiting attack traffic in a domain basis, and leave how those traffic is handled at their origin as the responsibility of corresponding domains. We will extend our

scheme to be applicable within a (source) domain so that attack sources are located and their traffic is confined more precisely at their origin.

Reliability of Control Messages. The authenticity as well as the integrity of control messages must be guaranteed in our collaborative defense mechanism. In addition, reliability on the requested messages (e.g., fair bandwidth allocation) is required for the recipient to take requested actions. We will design a control-message validation mechanism, whereby a recipient of control messages can ascertain whether a valid control request has been made by the message sender (i.e., router controller of a flooded domain).

BIBLIOGRAPHY

- [ABF⁺08] David G. Andersen, Hari Balakrishnan, Nick Feamster, Teemu Koponen, Daekyeong Moon, and Scott Shenker. Accountable Internet Protocol (AIP). In *Proc. ACM SIGCOMM*, Seattle, WA, August 2008.
- [ABKM01] David Andersen, Hari Balakrishnan, Frans Kaashoek, and Robert Morris. Resilient overlay networks. In *SOSP '01: Proceedings of the eighteenth ACM symposium on Operating systems principles*, pages 131–145, New York, NY, USA, 2001. ACM.
- [AC04] Katerina Argyraki and David R. Cheriton. Loose source routing as a mechanism for traffic policies. In *FDNA '04: Proceedings of the ACM SIGCOMM workshop on Future directions in network architecture*, pages 57–64, New York, NY, USA, 2004. ACM.
- [AC05a] Katerina Argyraki and David R. Cheriton. Active internet traffic filtering: real-time response to denial-of-service attacks. In *ATEC '05: Proceedings of the annual conference on USENIX Annual Technical Conference*, Berkeley, CA, USA, 2005. USENIX Association.
- [AC05b] Katerina Argyraki and David R. Cheriton. Network Capabilities: The Good, the Bad and the Ugly. *HotNets IV*, 2005.
- [AKM04] Guido Appenzeller, Isaac Keslassy, and Nick McKeown. Sizing router buffers. In *SIGCOMM*, 2004.
- [And03] David G. Andersen. Mayday: distributed filtering for internet services. In *USITS'03: Proceedings of the 4th conference on USENIX Symposium on Internet Technologies and Systems*, Berkeley, CA, USA, 2003. USENIX Association.
- [ANL00] Tuomas Aura, Pekka Nikander, and Jussipekka Leiwo. DOS-Resistant Authentication with Client Puzzles. In *Security Protocols Workshop*, pages 170–177, 2000.
- [ARW03] T. Anderson, T. Roscoe, and D. Wetherall. Preventing Internet denial-of-service with capabilities. In *Proc. of Hotnets-II*, 2003.
- [Bel00] Steven M. Bellovin. <http://www.cs.columbia.edu/smb/papers/draft-bellovin-itrace-00.txt>. 2000.
- [bgp] http://www.cisco.com/en/US/tech/tk365/technologies_tech_note09186a008-0094431.shtml.
- [bot] <http://www.computerworld.com/s/article/9076278/>.
- [CG05] Martin Casado and Tal Garfinkel. Opportunistic measurement: Extracting insight from spurious traffic. In *HotNets*, 2005.

- [cis] http://www.cisco.com/en/US/docs/ios/12_2sr/12_2srb/feature/guide/srmtrdoc.html.
- [CJB08] Zesheng Chen, Chuanyi Ji, and Paul Barford. Spatial-temporal characteristics of internet malicious sources. In *Proceedings of IEEE INFOCOM*, April, 2008.
- [CLSS08] Jerry Chou, Bill Lin, Subhabrata Sen, and Oliver Spatscheck. Proactive surge protection: a defense mechanism for bandwidth-based attacks. In *Proc. of the 17th USENIX Security symposium*, 2008.
- [CW77] J. Lawrence Carter and Mark N. Wegman. Universal classes of hash functions (extended abstract). In *STOC '77: Proceedings of the ninth annual ACM symposium on Theory of computing*, pages 106–112, 1977.
- [DZL06] David Dagon, Cliff Zou, and Wenke Lee. Modeling Botnet Propagation Using Time Zone. *Network and Distributed System Security Symposium*, 2006.
- [Fer00] P. Ferguson. Network Ingress Filtering: Defeating Denial of Service Attacks which employ IP Source Address Spoofing. *RFC 2827*, 2000.
- [FKSS01] Wu Chang Feng, Dilip D. Kandlur, Debanjan Saha, and Kang G. Shin. Stochastic Fair Blue: A Queue Management Algorithm for Enforcing Fairness. In *INFOCOM*, 2001.
- [Gli05] Virgil D. Gligor. Guaranteeing Access in Spite of Distributed Service-Flooding Attacks. In *LNCS*, volume 3364, pages 97–105. Springer Berlin / Heidelberg, 2005.
- [Goo02] M. T. Goodrich. Efficient Packet Marking for Large-Scale IP Traceback. pages 117–126, 2002.
- [GP01] Thomer M. Gil and Massimiliano Poletto. MULTOPS : a data-structure for denial-of-service attack detection. *10th USENIX Security Symposium*, 2001.
- [IB02] John Ioannidis and Steven M. Bellovin. Implementing Pushback: Router-Based Defense Against DDoS Attacks. In *NDSS*, 2002.
- [KK03] Aleksandar Kuzmanovic and Edward W. Knightly. Low-rate tcp-targeted denial of service attacks: the shrew vs. the mice and elephants. In *SIGCOMM '03*, pages 75–86, New York, NY, USA, 2003. ACM.
- [KKDC05] Srikanth Kandula, Dina Katabi, Bruce Davie, and Anna Charny. Walking the tightrope: responsive yet stable traffic engineering. In *SIGCOMM '05: Proceedings of the 2005 conference on Applications, technologies, architectures, and protocols for computer communications*, pages 253–264, New York, NY, USA, 2005. ACM.

- [KMR02] Angelos D. Keromytis, Vishal Misra, and Dan Rubenstein. SOS: secure overlay services. *SIGCOMM Comput. Commun. Rev.*, 32(4):61–72, 2002.
- [Li 98] Li Fan and Pei Cao and Jussara Almeida and Andrei Z. Broder. Summary cache: A scalable wide-area Web cache sharing protocol. In *IEEE/ACM Transactions on Networking*, pages 254–265, 1998.
- [LYL08] Xin Liu, Xiaowei Yang, and Yanbin Lu. To filter or to authorize: network-layer dos defense against multimillion-node botnets. In *SIGCOMM '08: Proceedings of the ACM SIGCOMM 2008 conference on Data communication*, pages 195–206, New York, NY, USA, 2008. ACM.
- [MBF⁺02] Ratul Mahajan, Steven M. Bellovin, Sally Floyd, John Ioannidis, Vern Paxson, and Scott Shenker. Controlling high bandwidth aggregates in the network. *Comput. Commun. Rev.*, 32(3), 2002.
- [MEFV08] Murtaza Motiwala, Megan Elmore, Nick Feamster, and Santosh Vempala. Path splicing. In *SIGCOMM*, pages 27–38, 2008.
- [MFW01] Ratul Mahajan, Sally Floyd, and David Wetherall. Controlling high-bandwidth flows at the congested router. In *ICNP '01*, Washington DC, 2001.
- [MPR02] Jelena Mirkovic, Gregory Prier, and Peter L. Reiher. Attacking DDoS at the Source. In *ICNP '02: Proceedings of the 10th IEEE International Conference on Network Protocols*, pages 312–321, Washington, DC, USA, 2002. IEEE Computer Society.
- [Par92] Craig Partridge. A Proposed Flow Specification. *RFC 1363*, 1992.
- [Pau90] Paul E. McKenney. Stochastic Fairness Queueing. In *INFOCOM*, pages 733–740, 1990.
- [PMR⁺07] P. Psenak, S. Mirtorabi, A. Roy, L. Nguyen, and P. Pillay-Esnault. Rfc-4915: Multi-topology (mt) routing in ospf. 2007.
- [PPP00] Rong Pan, Balaji Prabhakar, and Konstantinos Psounis. Choke, a stateless active queue management scheme for approximating fair bandwidth allocation. In *INFOCOM*, pages 942–951, 2000.
- [PWS⁺07] Bryan Parno, Dan Wendlandt, Elaine Shi, Adrian Perrig, and Bruce Maggs Yih-Chun Hu. Portcullis : Protecting Connection Setup from Denial-of-Capability Attacks. In *ACM SIGCOMM 2007*, Kyoto, Japan, August 2007.
- [RL95] Y. Rekhter and T. Li. A Border Gateway Protocol 4 (BGP-4). *RFC 1771*, 1995.

- [RMT06] Moheeb Abu Rajab, Fabian Monrose, and Andreas Terzis. On the impact of dynamic addressing on malware propagation. In *WORM '06*, 2006.
- [Rus03] Russ White. Securing BGP through Secure Origin BGP. *The Internet Protocol Journal*, 2003.
- [RZMT06] Moheeb Abu Rajab, Jay Zarfoss, Fabian Monrose, and Andreas Terzis. A Multifaced Approach to Understanding the Botnet Phenomenon. *Internet Measurement Conference*, 2006.
- [SAM07] Daniel R. Simon, Sharad Agarwal, and David A. Maltz. AS-Based Accountability as a Cost-effective DDoS Defense. *HotBots '07*, 2007.
- [SCM⁺05] Angelos Stavrou, Debra L. Cook, William G. Morein, Angelos D. Keromytis, Vishal Misra, and Dan Rubenstein. WebSOS: An Overlay-based System For Protecting Web Servers From Denial of Service Attacks. *Journal of Communication Networks*, 48(5), August 2005.
- [Shr95] Shreedhar,, M. and Varghese,, George. Efficient fair queueing using deficit round robin. In *SIGCOMM '95: Proceedings of the conference on Applications, technologies, architectures, and protocols for computer communication*, pages 231–242, New York, NY, USA, 1995. ACM.
- [SK05] Angelos Stavrou and Angelos D. Keromytis. Countering DoS attacks with stateless multipath overlays. In *CCS '05: Proceedings of the 12th ACM conference on Computer and communications security*, pages 249–259, New York, NY, USA, 2005. ACM Press.
- [SP01] Dawn Xiaodong Song and Adrian Perrig. Advanced and Authenticated Marking Schemes for IP Traceback. In *INFOCOM*, pages 878–886, 2001.
- [SP09] Ahren Studer and Adrian Perrig. The coremelt attack. In *ESORICS*, Saint Malo, France, September 2009.
- [SPW02] Stuart Staniford, Vern Paxson, and Nicholas Weaver. How to Own the Internet in Your Spare Time. In *Proceedings of the 11th USENIX Security Symposium*, 2002.
- [SSZ98] Ion Stoica, Scott Shenker, and Hui Zhang. Core -Stateless Fair Queueing: Achieving Approximately Fair Bandwidth Allocations in High Speed Networks. In *SIGCOMM*, 1998.
- [Ste00] Stephen Kent and Charles Lynn and Joanne Mikkelsen and Karen Seo. Secure Border Gateway Protocol (S-BGP). *IEEE Journal on Selected Areas in Communications*, 18:103–116, 2000.

- [SWKA00] Stefan Savage, David Wetherall, Anna R. Karlin, and Tom Anderson. Practical network support for IP traceback. In *SIGCOMM*, pages 295–306, 2000.
- [vABHL03] L. von Ahn, M. Blum, N. Hopper, and J. Langford. CAPTCHA: Using hard AI problems for security. In *Proceedings of Eurocrypt*, pages 294–311, 2003.
- [WR03] Xiaofeng Wang and Michael K. Reiter. Defending Against Denial-of-Service Attacks with Puzzle Auctions. In *SP '03: Proceedings of the 2003 IEEE Symposium on Security and Privacy*, pages 78–92, Washington, DC, USA, 2003. IEEE Computer Society.
- [WVB⁺06] Michael Walfish, Mythili Vutukuru, Hari Balakrishnan, David Karger, and Scott Shenker. DDoS Defense by Offense. In *ACM SIGCOMM 2006*, Pisa, Italy, September 2006.
- [XG05] Ying Xu and Roch Guérin. On the robustness of router-based denial-of-service (dos) defense systems. *Comput. Commun. Rev.*, 35(3), 2005.
- [XG06] Ying Xu and Roch Guérin. A double horizon defense design for robust regulation of malicious traffic. *SecureComm*, 2006.
- [XR06] Wen Xu and Jennifer Rexford. Miro: multi-path interdomain routing. In *SIGCOMM '06: Proceedings of the 2006 conference on Applications, technologies, architectures, and protocols for computer communications*, pages 171–182, New York, NY, USA, 2006. ACM Press.
- [YPS03] A. Yaar, A. Perrig, and D. Song. Pi: A Path Identification Mechanism to Defend against DDoS Attacks. In *IEEE Symposium on Security and Privacy*, 2003.
- [YPS04] Abraham Yaar, A. Perrig, and D. Song. SIFF: A Stateless Internet Flow Filter to Mitigate DDoS Flooding Attacks. In *Proceedings of the IEEE Security and Privacy Symposium*, 2004.
- [YW06] Xiaowei Yang and David Wetherall. Source selectable path diversity via routing deflections. In *SIGCOMM '06: Proceedings of the 2006 conference on Applications, technologies, architectures, and protocols for computer communications*, pages 159–170, New York, NY, USA, 2006. ACM.
- [YWA05] Xiaowei Yang, David Wetherall, and Thomas Anderson. A DoS-limiting network architecture. In *SIGCOMM '05*, 2005.
- [YWA08] Xiaowei Yang, David Wetherall, and Thomas Anderson. A DoS-limiting network architecture. In *IEEE/ACM TRANSACTIONS ON NETWORKING*, 2008.