Controlling IP Spoofing based DDoS Attacks Through Inter-Domain Packet Filters*

Zhenhai Duan†, Xin Yuan, and Jaideep Chandrashekar

Abstract

The Distributed Denial of Services (DDoS) attack is a serious threat to the legitimate use of the Internet. Prevention mechanisms are thwarted by the ability of attackers to forge, or spoof, the source addresses in IP packets. By employing IP spoofing, attackers can evade detection and put a substantial burden on the destination network for policing attack packets. In this paper we propose an inter-domain packet filter (IDPF) architecture that can mitigate the level of IP spoofing on the Internet. IDPFs are constructed from the information implicit in BGP route updates and are deployed in network border routers. A key feature of the scheme is that it does not require global routing information. In this paper we study the conditions under which the IDPF framework works correctly in that it does not discard packets with valid source addresses. Based on extensive simulation studies, we show that even with partial deployment on the Internet, IDPFs can proactively limit the spoofing capability of attackers. In addition, they can help localize the origin of an attack packet to a small number of candidate networks.

Keywords: IP Spoofing, DDoS, BGP, Network-level Security and Protection, Routing Protocols

1 Introduction

Distributed Denial of Service (DDoS) attacks pose an increasingly grave threat to the Internet, as evidenced by recent DDoS attacks mounted on both popular Internet sites [12] and the Internet infrastructure [11]. Alarmingly, DDoS attacks are observed on a daily basis on most of the large backbone networks [26]. One of the factors that complicate the mechanisms for policing such attacks is *IP spoofing*, the act of forging the source addresses in IP packets. By masquerading as a different

*A preliminary version of this paper appeared in the Proc. IEEE INFOCOM 2006 with the title "Constructing Inter-Domain Packet Filters to Control IP Spoofing Based on BGP Updates".

Zhenhai Duan and Xin Yuan are with the Computer Science Department, Florida State University, Tallahassee, FL 32306 USA (email: {duan,xyuan}@cs.fsu.edu).

Jaideep Chandrashekar is with the Intel Research/CTL, Santa Clara, CA 95054 USA (email: jaideep.chandrashekar@intel.com)

†Corresponding author.

host, an attacker can hide its actual identity and location, rendering source-based packet filtering less effective. It has been shown that a large part of the Internet is vulnerable to IP spoofing [3, 4].

Recently, there is anecdotal evidence of attackers to stage attacks utilizing *bot-nets*[1] [24]. In this case, since the attacks are carried out through intermediaries, i.e., the compromised "bots", it is tempting to believe that the use of IP spoofing is less of a factor than previously. However, recent studies present evidence to the contrary and show that IP spoofing is still a commonly observed phenomenon [29, 31].

It is our contention that IP spoofing will remain popular for a number of reasons. First, IP spoofing makes it harder to isolate attack traffic from legitimate traffic—packets with spoofed source addresses may appear to be from all around the Internet. Second, it presents the attacker with an easy way to insert a level of indirection, which shifts the burden to the victim; substantial effort is required to localize the source of the attack traffic [2, 13, 35, 36]. Finally, many popular attacks use IP spoofing and require the ability to forge source addresses. Man-in-the-middle attacks, such as variants of TCP hijack and DNS poisoning attacks [10, 38], are carried out by the attacker masquerading as the host at the other end of a valid transaction. Reflector-based attacks use IP spoofing to masquerade as some victim host that contacts a number of hosts, resulting in the victim being flooded by replies from all these hosts [33]. TCP SYN flood attacks rely on spoofing addresses of hosts that are unable to respond to replies [6]. These factors indicate that IP spoofing is unlikely to decrease in the near future.

Although attackers can insert arbitrary source addresses into IP packets, they cannot, however, control the actual paths that the packets take to the destination. Based on this observation, Park and Lee [32] proposed the *route-based packet filters* as a way to mitigate IP spoofing. The intuition in this scheme is that, assuming single-path routing, there is exactly one single path $p(s, d)$ between source node s and destination node d. Hence, any packets with source address s and destination address d that appear in a router not in $p(s, d)$ should be discarded. However, constructing a specific route-based packet filter in a node requires the knowledge of global routing decisions made by *all* the other nodes in the network, which is hard to reconcile on the current BGP-based Internet

[1] collections of hundreds or thousands of compromised hosts, "recruited" by worm or virus infection [9].

routing infrastructure [21, 34, 37].

The current Internet consists of approximately 15,000 network domains or autonomous systems (ASes), each of which is a logical collection of networks with common administrative control. Each AS communicates with its neighbors using the Border Gateway Protocol (BGP), the de-facto inter-domain routing protocol, to exchange information about its own networks and others that it can reach [21, 34, 37]. BGP is a *policy-based* routing protocol in that both the selection and the propagation of the best route to reach a destination at an AS are guided by some locally defined routing policies. Given the insular nature of how policies are applied at individual ASes, it is impossible for an AS to acquire the complete knowledge of routing decisions made by all the other ASes. Hence constructing route-based packet filters as proposed in [32] is an open challenge in the current Internet routing regime.

Inspired by the idea of route-based packet filters, we propose an Inter-Domain Packet Filter (IDPF) architecture. The IDPF architecture takes advantage of the fact that while network connectivity may imply a large number of *potential* paths between source and destination domains, commercial relationships between ASes act to restrict to a much smaller set the number of *feasible* paths that can be used to carry traffic from the source to the destination [15]. In this paper we focus our attention on the construction of IDPFs based solely on locally exchanged BGP updates. We will investigate how other AS relationship and routing information may help further improve the performance of IDPFs in our future work.

We show that locally exchanged routing information between neighbors, i.e., BGP route updates, is sufficient to identify feasible paths and construct IDPFs, assuming all ASes on the Internet employ a set of routing policies that are commonly used today [18, 19, 22]. Like route-based packet filters [32], the proposed IDPFs cannot stop all spoofed packets. However, when spoofed packets are not filtered out, IDPFs can help localize the origin of attack packets to a small set of ASes, which can significantly improve the IP traceback situation [2, 13, 35, 36]. We summarize the key contributions of this paper in the following:

1) We describe how to practically construct inter-domain packet filters *locally* at an AS by using only the BGP route updates being exchanged between the AS and its immediate neighbors.

2) We study the conditions under which the proposed IDPF framework works correctly in that it will not discard packets with valid source addresses.
3) To evaluate the effectiveness of the architecture, we conduct extensive simulation studies based on AS topologies and AS paths extracted from real BGP data provided by the Route-Views project [30]. Our results show that, even with partial deployment, the architecture can proactively limit an attacker's ability to spoof packets. When a spoofed packet cannot be stopped, IDPFs can help localize the attacker to a small number of candidate ASes, reducing the effort and increasing the accuracy of IP traceback schemes.
4) We show that unlike some protection schemes that provide intangible local benefits for deployment, the IDPF architecture provides better protection against IP spoofing based DDoS attacks on local networks, which presents incentives for network operators to deploy IDPFs.

The rest of this paper is organized as follows. We discuss related work in Section 2. We provide an abstract model of BGP in Section 3. Section 4 presents the IDPF architecture. Section 5 discusses practical deployment issues. We report our simulation study of IDPFs in Section 6. We conclude the paper and discuss future work in Section 7.

2 Related Work

The idea of IDPF is motivated by the work carried out by Park and Lee [32], which was the first effort to evaluate the relationship between topology and the effectiveness of route-based packet filtering. The authors showed that packet filters that are constructed based on the *global* routing information can significantly limit IP spoofing when deployed in just a small number of ASes. In this work, we extend the idea and demonstrate that filters that are built based on *local* BGP updates can also be effective.

Unicast reverse path forwarding (uRPF) [1] requires that a packet is forwarded only when the interface that the packet arrives on is exactly the same used by the router to reach the source IP of the packet. If the interface does not match, the packet is dropped. While simple, the scheme is limited given that Internet routing is inherently asymmetric, i.e., the forward and reverse paths between a pair of hosts is often quite different. In Hop-Count Filtering (HCF) [23], each end system

maintains a mapping between IP address aggregates and valid hop counts from the origin to the end system. Packets that arrive with a different hop count are suspicious and are therefore discarded or marked for further processing. In [27], Li *et al.*, described SAVE, a new protocol for networks to propagate valid network prefixes along the same paths that data packets will follow. Routers along the paths can thus construct the appropriate filters using the prefix and path information. Bremler-Barr and Levy proposed a spoofing prevention method (SPM) [5], where packets exchanged between members of the SPM scheme carry an authentication key associated with the source and destination AS domains. Packets arriving at a destination domain with an invalid authentication key (w.r.t. the source domain) are spoofed packets and are discarded.

In the Network Ingress Filtering proposal described in [16], traffic originating from a network is forwarded only if the source IP in the packets is from the network prefix belonging to the network. Ingress filtering primarily prevents a specific network from being used to attack others. Thus, while there is a collective social benefit in everyone deploying it, individuals do not receive direct incentives. Finally, the Bogon Route Server Project [39] maintains a list of *bogon* network prefixes that are not routable on the public Internet. Examples include private RFC 1918 address blocks and unassigned address prefixes. Packets with source addresses in the bogon list are filtered out. However, this mechanism cannot filter out attack packets carrying routable but spoofed source addresses.

3 Border Gateway Protocol and AS Interconnections

In this section, we briefly describe a few key aspects of BGP that are relevant to this paper (see [37] for a comprehensive description). To begin with, we model the AS graph of the Internet as an *undirected* graph $G = (V, E)$. Each node $v \in V$ corresponds to an Autonomous System (AS), and each edge $e(u, v) \in E$ represents a BGP session between two neighboring ASes $u, v \in V$. To simplify the exposition, we assume that there is at most one edge between neighboring ASes.[2]

Each node owns one or multiple network prefixes. Nodes exchange BGP route updates, which may be announcements or withdrawals, to learn of changes in reachability to destination network

[2]This is merely for convenience; the simplification does not affect the correctness of our scheme.

prefixes. A route withdrawal, containing a list of network prefixes, indicates that the sender of the withdrawal message can no longer reach the prefixes. In contrast, a route announcement indicates that the sender knows of a path to a network prefix. The route announcement contains a list of *route attributes* associated with the destination network prefix. Of particular interest to us are the path vector attribute, `as_path`, which is the sequence of ASes that this route has been propagated over, and the `local_pref` attribute that describes the *degree of local preference* associated with the route. We will use `r.as_path`, `r.local_pref`, and `r.prefix` to denote the `as_path` attribute, the `local_pref`, and the destination network prefix of r, respectively. Let $r.\texttt{as\_path} = \langle v_k v_{k-1} \dots v_1 v_0 \rangle$. The route was originated (first announced) by node v_0, which owns the address space described by `r.prefix`. Before arriving at node v_k, the route was carried over nodes $v_1, v_2, \dots, v_{k-1}$ in that order. For $i = k, k-1, \dots, 1$, we say that edge $e(v_i, v_{i-1})$ is on the AS path, or $e(v_i, v_{i-1}) \in r.\texttt{as\_path}$.

When there is no confusion, route r and its AS path $r.\texttt{as\_path}$ are used interchangeably. For convenience, we also consider a specific destination AS d; all route announcements and withdrawals are specific to the network prefixes owned by d. For simplicity, notation d is also used to denote the network prefixes owned by the AS d. As a consequence, a route r that can be used to reach the network prefixes owned by destination d may simply be expressed as a route to *reach destination* d.

3.1 Policies and Route Selection

Each node only selects and propagates to neighbors a single *best* route to the destination, if any. BGP is a *policy-based* routing protocol in that both the selection and the propagation of best routes are guided by locally defined routing policies. Two distinct sets of routing policies are normally employed by a node: *import* policies and *export* policies. Neighbor-specific import policies are applied upon routes learned from neighbors, whereas neighbor-specific export policies are imposed on locally-selected best routes before they are propagated to the neighbors. Both the import routing policies and the export routing policies employed by an AS are largely determined by the relationships between the AS and its neighbors, which we will discuss in detail in the next subsection. In this subsection we simply assume the existence of the import and export routing policies.

In general, *import* policies can affect the "desirability" of routes by modifying route attributes

such as the local preference attribute `local_pref` (Section 3.2). Let r be a route (to destination d) received at v from node u. We denote by $\mathbf{import}(v \leftarrow u)[\{r\}]$ the possibly modified route that has been *transformed* by the import policies. After the routes are passed through the import policies at node v, they are stored in v's routing table. The set of all such routes is denoted as $\mathbf{candidateR}(v, d)$:

$$\mathbf{candidateR}(v, d) = \{r : \mathbf{import}(v \leftarrow u)[\{r\}] \neq \{\}, r.\mathbf{prefix} = d, \forall u \in N(v)\}. \quad (1)$$

Here, $N(v)$ is the set of v's neighbors.

Among the set of candidate routes $\mathbf{candidateR}(v, d)$, node v selects a single best route to reach the destination based on a well defined procedure (see [8]). To aid in description, we shall denote the outcome of the selection procedure at node v, i.e., the best route, as $\mathbf{bestR}(v, d)$, which reads the *best route to destination d at node v.*

Having selected $\mathbf{bestR}(v, d)$ from $\mathbf{candidateR}(v, d)$, v then exports the route to its neighbors after applying neighbor specific *export policies*. The export policies determine if a route should be forwarded to the neighbor, and if so, modify the route attributes according to the policies (Section 3.2). We denote by $\mathbf{export}(v \rightarrow u)[\{r\}]$ the route sent to neighbor u by node v, after node v applies the export policies on route r.

BGP is an incremental protocol: updates are generated only in response to network events. In the absence of any events, no route updates are triggered or exchanged between neighbors, and we say that the routing system is in a stable state. Formally,

Definition 1 (Stable Routing State) *A routing system is in a stable state if all the nodes have selected a best route to reach other nodes and no route updates are generated (and propagated) by any node.*

3.2 AS Relationships and Routing Policies

The specific routing policies that an AS employs internally is largely determined by economics: connections between ASes follow a few commercial relations. A pair of ASes can enter into one of the following arrangements [18, 22]:

- *provider-customer:* In this kind of arrangement, a customer AS pays the provider AS to carry its traffic to the rest of the Internet. This arrangement is the most common and is natural when the provider is much larger in size than the customer.

- *peer-peer:* In a mutual peering agreement, the ASes decide to carry traffic from each other (and their customers). This is only natural when the traffic from each other is roughly balanced. Mutual peers do not carry transit traffic for each other.

- *sibling-sibling:* In this type of arrangement, two ASes provide mutual transit service to each other (often as backup connectivity or for reasons of economy). Each of the two sibling ASes can be regarded as the provider of the other AS.

An AS's relationship with a neighbor largely determines the neighbor-specific import and export routing policies the AS employs on the routes propagated between the two ASes. In this paper we assume that each AS on the Internet sets its import routing policies and export routing policies according to the rules specified in Table 1 [19] and Table 2 [18, 22], respectively. These rules are commonly used by ASes on the current Internet. In Table 1, r_1 and r_2 denote the routes (to destination d) received by node v from neighbors u_1 and u_2, respectively; and $customer(v)$, $peer(v)$, $provider(v)$, and $sibling(v)$ denote the set of customers, peers, providers, and siblings of node v, respectively. The import routing policies in Table 1 state that an AS will prefer the routes learned from customers or siblings over the routes learned from peers or providers.

In Table 2, the columns marked with **r1-r4** specify the export policies employed by an AS to announce routes to providers, customers, peers, and siblings, respectively. For instance, export rule **r1** instructs that an AS will announce routes to its own networks, and routes learned from customers and siblings to a provider, but it will not announce routes learned from other providers and peers to the provider. The net effect of these rules is that they limit the possible paths between each pair of ASes. The routing policies described in Tables 1 and 2 are not complete. In a few cases, ASes may choose to apply less restrictive policies to satisfy traffic engineering goals. For the moment, we assume that all ASes follow the import and export routing policies specified in Tables 1 and 2 and that each AS accepts legitimate routes exported by neighbors. More general cases will be discussed

at the end of the next section.

if $((u_1 \in customer(v) \cup sibling(v))$ and $(u_2 \in peer(v) \cup provider(v)))$ then r_1.local_pref $> r_2$.local_pref

Table 1: Import routing policies at an AS.

Export rules		**r1**	**r2**	**r3**	**r4**
Export routes to		provider	customer	peer	sibling
Learned from	provider	**no**	**yes**	**no**	**yes**
	customer	**yes**	**yes**	**yes**	**yes**
	peer	**no**	**yes**	**no**	**yes**
	sibling	**yes**	**yes**	**yes**	**yes**
Own routes		**yes**	**yes**	**yes**	**yes**

Table 2: Export routing policies at an AS.

If AS b is a provider of AS a, and AS c is a provider of AS b, we call c an *indirect* provider of a, and a an *indirect* customer of c. Indirect siblings are defined in a similar fashion. The import and export routing policies in Tables 1 and 2 imply that an AS will distribute the routes to direct or indirect customers/siblings to its peers and providers. If $e(u, v) \in \mathbf{bestR}(s, d)$.as_path, we say that u is the best upstream neighbor of node v for traffic from node s to destination d, and denote u as $u = \mathbf{bestU}(s, d, v)$. For ease of exposition, we augment the AS graph with the relationships between neighboring ASes. We refer to an edge from a provider to a customer AS as a provider-to-customer edge, an edge from a customer to provider as a customer-to-provider edge, and an edge connecting sibling (peering) ASes as sibling-to-sibling (peer-to-peer) edge. A *downhill* path is a sequence of edges that are either provider-to-customer or sibling-to-sibling edges, and an *uphill path* is a sequence of edges that are either customer-to-provider or sibling-to-sibling edges. Gao [18] established the following theorem about the candidate routes in a BGP routing table.

Theorem 1 (Gao [18]) *If all ASes set their export policies according to* **r1-r4**, *any candidate route in a BGP routing table is either (a) an uphill path, (b) a downhill path, (c) an uphill path followed by a downhill path, (d) an uphill path followed by a peer-to-peer edge, (e) a peer-to-peer*

edge followed by a downhill path, or (f) an uphill path followed by a peer-to-peer edge, which is followed by a downhill path.

4 Inter Domain Packet Filters

In this section we discuss the intuition behind the IDPF architecture, describe how IDPFs are constructed using BGP route updates, and establish the correctness of IDPFs. After that, we discuss the case where ASes have routing policies that are less restrictive than the ones in Tables 1 and 2. We shall assume that the routing system is in the *stable routing state* in this section. We will discuss how IDPFs fare with network routing dynamics in the next section.

Let $M(s, d)$ denote a packet whose source address is s (or more generally, the address belongs to AS s), and destination address d. A packet filtering scheme decides whether a packet should be forwarded or dropped based on certain criteria. One example is the route-based packet filtering [32]:

Definition 2 (Route-Based Packet Filtering) *Node v accepts packet $M(s, d)$ forwarded from node u if and only if $e(u, v) \in \mathbf{bestR}(s, d)$. Otherwise, the source address of the packet is spoofed, and the packet is discarded by v.*

In the context of preventing IP spoofing, an ideal packet filter should discard spoofed packets while allowing legitimate packets to reach the destinations. Since even with the perfect routing information the route-based packet filters cannot identify all spoofed packets [32], a valid packet filter should focus on not dropping any legitimate packets. Accordingly, we define the correctness of a packet filter as follows.

Definition 3 (Correctness of Packet Filtering) *A packet filter is* correct *if it does not discard packets with valid source addresses when the routing system is stable.*

Clearly, the route-based packet filtering is correct, because valid packets from source s to destination d will only traverse the edges on $\mathbf{bestR}(s, d)$ when the routing system is stable.

4.1 Motivating IDPFs

Although route-based packet filtering is correct, it requires each node to have the global knowledge of $\mathbf{bestR}(s, d)$. In the current Internet architecture that uses BGP as the inter-domain routing protocol, such information is not available. In BGP, route selection is a local decision, i.e., s computes $\mathbf{bestR}$(s,d) based on the set of routes in its routing table and preferences that an operator in AS s has defined. This information may not be available at nodes in $\mathbf{bestR}$(s,d). Consequently, route-based packet filtering cannot be applied in the current BGP-based Internet routing regime.

IDPF overcomes this problem by using the information implicit in BGP updates to construct the filters. We use the following concepts to illustrate the idea of IDPF. A *topological route* between nodes s and d is a loop-free path between the two nodes. Topological routes are implied by the network connectivity. A *topological route* is a *feasible route* under BGP if and only if the construction of the route does not violate the routing policies imposed by the commercial relationship between ASes (Tables 1 and 2). Formally, let $\mathbf{feasibleR}(s, d)$ denote the set of feasible routes from s to d, then $\mathbf{feasibleR}(s, d)$ can be recursively defined as follows:

$$\mathbf{feasibleR}(s, d) =$$
$$\{\langle s \oplus \cup_{u\,:\,\mathbf{import}(s \leftarrow u)[\{r\}] \neq \{\}, r.\texttt{prefix} = d, u \in N(s)} \mathbf{feasibleR}(u, d)\rangle\}, \qquad (2)$$

where $\oplus$ is the concatenation operation, e.g., $\{s \oplus \{\langle ab\rangle, \langle uv\rangle\}\} = \{\langle sab\rangle, \langle suv\rangle\}$. Notice that $\mathbf{feasibleR}(s, d)$ contains *all* the routes between the pair that does not violate the import and export routing policies specified in Tables 1 and 2. Obviously, $\mathbf{bestR}(s, d) \in \mathbf{candidateR}(s, d) \subseteq \mathbf{feasibleR}(s, d)$. Each of the feasible routes can potentially be a candidate route in a BGP routing table. Theorem 1 also applies to feasible routes.

Definition 4 (Feasible Upstream Neighbor) *Consider a feasible route $r \in \mathbf{feasibleR}(s, d)$. If an edge $e(u, v)$ is on the feasible route, i.e., $e(u, v) \in r.$*`as_path`*, we say that node u is a feasible upstream neighbor of node v for packet $M(s, d)$. The set of all such feasible upstream neighbors of v (for $M(s, d)$) is denoted as $\mathbf{feasibleU}(s, d, v)$.*

The intuition behind the IDPF framework is the following. First, it is possible for a node v

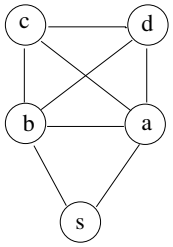
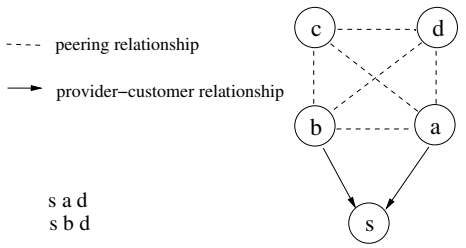


Figure 1: An example network topology.

s a d
s b d
s a b d
s a c d
s b a d
s b c d
s a b c d
s a c b d
s b a c d
s b c a d

(a) Topological routes implied by connectivity



s a d
s b d

(b) Feasible routes constrained by routing policies

Figure 2: Routes between source s and destination d.

to infer its feasible upstream neighbors using BGP route updates. The technique to infer feasible upstream neighbors is described in the next sub-section. Since $\mathbf{bestR}(s,d) \in \mathbf{candidateR}(s,d) \subseteq \mathbf{feasibleR}(s,d)$, a node can only allow $M(s,d)$ from its feasible upstream neighbors to pass and discard all other packets. Such a filtering will not discard packets with valid source addresses. Second, although network connectivity (topology) may imply a large number of topological routes between a source and destination, commercial relationship between ASes and routing policies employed by ASes act to restrict the size of $\mathbf{feasibleR}(s,d)$. Consider the example in Fig. 1. Figs. 2(a) and (b) present the topological routes implied by network connectivity and feasible routes constrained by routing policies between source s and destination d, respectively. In Fig. 2(b) we assume that nodes a, b, c, and d have mutual peering relationship, and that a and b are providers to s. We see that although there are 10 topological routes between source s and destination d, we only have 2 feasible routes that are supported by routing policies. Of more importance to IDPF is that, although network topology may imply all neighbors can forward a packet allegedly from a source to a node, feasible routes constrained by routing policies help limit the set of such neighbors. As an example, let us consider the situation at node d. Given that only nodes a and b (but not c) are on the feasible routes from s to d as node d concerns, node d can infer that all packets forwarded by node c and allegedly from source s are spoofed and should be discarded.

It is clear that packet filters based on feasible routes are less powerful than those based on best routes, given that $\mathbf{bestR}(s,d) \in \mathbf{candidateR}(s,d) \subseteq \mathbf{feasibleR}(s,d)$. On the other hand, AS relationships normally restrict the feasible routes between a pair of source and destination to

a small set, which makes feasible-route based packet filtering a practical and promising approach against IP spoofing. In the following subsection, we will present a mechanism for each node to identify the set of *feasible* upstream neighbors that can forward packet $M(s,d)$ to the node, based on locally exchanged BGP updates between the node and its immediate neighbors.

4.2 Constructing IDPFs

The following lemma summarizes the technique to identify the feasible upstream neighbors of node v for packet $M(s,d)$.

Lemma 2 *Consider a feasible route r between source s and destination d. Let $v \in r.$*`as_path` *and u be the feasible upstream neighbor of node v along r. When the routing system is stable,* $\mathbf{export}(u \rightarrow v)[\{\mathbf{bestR}(u,s)\}] \neq \{\}$*, assuming that all ASes follow the import and export routing policies in Tables 1 and 2 and that each AS accepts legitimate routes exported by neighbors.*

Lemma 2 states that if node u is a feasible upstream neighbor of node v for packet $M(s,d)$, node u must have exported to node v its best route to reach the *source* s.

Proof: Since Theorem 1 applies to feasible routes, a feasible route can be one of the six types of paths in Theorem 1. In the following we assume the feasible route r is of type (f), i.e., an uphill path followed by a peer-to-peer edge, which is followed by a downhill path. Cases where r has other types (a)-(e) can be similarly proved. To prove the lemma, we consider the possible positions of nodes u and v in the feasible route.

Case 1: Nodes u and v belong to the uphill path. Then node s must be an (indirect) customer or sibling of node u. From the import routing policies in Table 1 and the export routing policy **r1** and the definition of indirect customers/siblings, we know u will propagate to (provider) node v the reachability information of s.

Case 2: $e(u,v)$ is the peer-to-peer edge. This case can be similarly proved as case 1 (based on the import routing policies in Table 1 and the export routing policy **r3**).

Case 3: Nodes u and v belong to the downhill path. Let $e(x,y)$ be the peer-to-peer edge along the feasible route r, and note that u is an (indirect) customer of y. From the proof of case 2, we know

that node y learns the reachability information of s from x. From the export routing policy **r2** and the definition of indirect customers, node y will propagate the reachability information of s to node u, which will further export the reachability information of s to (customer) node v. ∎

It is critical to note that Lemma 2 only states that a feasible upstream neighbor u of neighbor v for packet $M(s,d)$ exports to v its best route to reach *source* s. Node v may choose a node other than u to reach s. Therefore, Lemma 2 does not imply the symmetry of best routes. For example, although feasible route $\langle s \dots uv \dots d \rangle$ may be the best route from s to d, route $\langle d \dots vu \dots s \rangle$ may *not* be the best route from d to s.

Relying on Lemma 2, a node can identify the feasible upstream neighbors for packet $M(s,d)$ and conduct inter-domain packet filtering as follows. Note that the filters are defined at a node specific to each neighbor.

Definition 5 (Inter-Domain Packet Filtering (IDPF)) *Node v will accept packet $M(s,d)$ forwarded by a neighbor node u, if and only if* $\mathbf{export}(u \rightarrow v)[\{\mathbf{bestR}(u,s)\}] \neq \{\}$.[3] *Otherwise, the source address of the packet must have been spoofed, and the packet should be discarded by node v.*

4.3 Correctness of IDPF

Theorem 3 *An IDPF as defined in* Definition *5 is correct.*

Proof: Without loss of generality, consider source s, destination d, and a node $v \in \mathbf{bestR}(s,d).\texttt{as\_path}$ such that v deploys an IDPF filter. In order to prove the correctness of the theorem, we need to establish that v will not discard packet $M(s,d)$ forwarded by the best upstream neighbor u, along $\mathbf{bestR}(s,d)$.

Recall from the best route selection process, the best route between a source and destination is also a feasible route between the two ($\mathbf{bestR}(s,d) \in \mathbf{candidateR}(s,d) \subseteq \mathbf{feasibleR}(s,d)$).

[3] As a technical detail, the condition should be $\mathbf{import}(v \leftarrow u)[\mathbf{export}(u \rightarrow v)[\{\mathbf{bestR}(u,s)\}] \neq \{\}$. That is, not only is $\mathbf{bestR}(u,s)$ exported to v by u, but also accepted by v. We ignore this technical detail for the clarity of our presentation.

Therefore, u is also a feasible upstream neighbor of node v for packet $M(s,d)$. From Lemma 2, u must have exported to node v its best route to source s. That is $\mathbf{export}(u \rightarrow v)[\{\mathbf{bestR}(u,s)\}] \neq \{\}$. From *Definition* 5, packet $M(s,d)$ forwarded by node u will not be discarded by v, and we have established the correctness of the theorem. ∎

4.4 Routing Policy Complications

As we have discussed in Section 3.2, the import routing policies and the export routing policies specified in Tables 1 and 2 are not complete. In particular, multi-homed ASes may employ less restrictive routing policies for traffic engineering or other purposes. In this section we first present two traffic engineering examples that do not follow the import and export routing policies specified in Tables 1 and 2. Then, we discuss how ASes employing these special traffic engineering practices should control the forwarding of their traffic to ensure the delivery of their traffic in the IDPF framework.

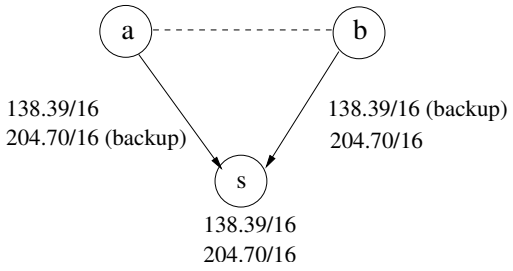


Figure 3: Automatic backup route.

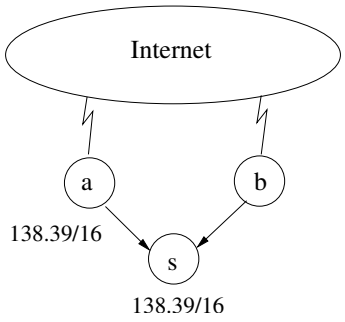


Figure 4: Conditional route advertisement.

In the first example (Figure 3), taken from [37], ASes a and b are providers of customer AS s, and s has two network prefixes `138.39/16` and `204.70/16`. The link between a and s is used as the primary link for `138.39/16` and backup link for `204.70/16`; while the link between b and s is used in a reverse manner. To achieve this traffic engineering and reliability goal, s informs a to assign the direct customer route r_1 between a and s a lower local preference over the peering route r_2 learned from b to reach the network prefix `204.70/16`. That is $r_1.\texttt{local\_pref} < r_2.\texttt{local\_pref}$. This local preference assignment at node a does not follow the import routing policies defined in

Table 1, which requires that an AS should prefer a direct route to a customer over an indirect route through a peer to reach a customer.

Now consider the second example shown in Figure 4. Customer s has a primary provider a and a backup provider b. s realizes this goal using a technique called conditional route advertisement; prefix `138.39/16` is announced to the backup provider b only if the link to the primary provider a fails. This does not follow the export routing policy **r1** defined in Table 2, which states that a customer will always export to its providers the routes to its own network prefixes.

It is critical to investigate if such local routing policies can be supported in the IDPF framework. Note that in both examples, the customer s controls the route propagation by affecting the local preference of the routes in providers (Figure 3) or by conditional route advertisement (Figure 4). As long as the customer AS does not forward packets through the backup route while the primary route is still available, the IDPF architecture will not discard any valid packets. This requirement is not hard to meet since the customer controls both the route propagation and traffic delivery. The same observation applies to other cases where the import routing policies and the export routing policies specified in Tables 1 and 2 are not followed. Therefore we have the following restricted traffic forwarding policy for the ASes that do not follow the import and export routing policies specified in Tables 1 and 2.

1. **Restricted traffic forwarding policy:** If an AS does not follow the import and export routing policies specified in Tables 1 and 2, the AS should restrict its traffic forwarding in a way that the AS will not forward traffic along the backup route as long as the primary route of the traffic is available.

If each AS on the Internet follows the import routing policies in Table 1 and the export routing policies in Table 2 and the restricted traffic forwarding policy, we can establish the correctness of IDPFs as defined in *Definition* 5 on the Internet. The proof is similar to the one of Lemma 2 and Theorem 3 and we omit it here.

5 Practical Deployment Issues of IDPFs

5.1 Incremental Deployment

From the description in Section 4, it should be clear that the IDPFs can be deployed independently in each AS. IDPFs are deployed at the border routers, so that IP packets can be inspected before they enter the network. We term border routers that are IDPF enabled as "IDPF nodes". An IDPF node is required to track the destination network prefixes announced by each neighbor. Typically, we expect that BGP speaking routers will support IDPF. In the case that an IDPF node is *not* a BGP router, it needs to obtain the corresponding prefix announcement information from the BGP speaking routers.[4]

When a packet arrives at an IDPF node, it needs to be associated with the specific neighbor that forwarded the packet. Subsequently, the IDPF matches the address against the set of prefixes announced by the specific neighbor. If a matching prefix exists, the packet is forwarded to the immediate destination in the AS or further routed towards the final destination. Otherwise, it is dropped by the IDPF node at the ingress of the network.

5.2 Handling Routing Dynamics

In the discussion so far, we have assumed that the AS graph is a static structure. However, in reality, the graph does change, triggering the generation of BGP updates and altering the paths that ASes use to reach each other. In this subsection, we examine how routing dynamics may affect the operation of IDPFs. We consider two different types of routing dynamics: 1) those caused by network failures; 2) and those caused by the creation of a new network (or recovery from a fail-down network event). Routing dynamics caused by routing policy changes can be similarly addressed and we omit them here.

Note that while filters are constructed based on route updates received from neighbors, they are completely oblivious to the specifics of the announced route. Moreover, the set of feasible upstream

[4]The simplest case to accomplish this would be to maintain BGP peering sessions with the BGP routers in the AS.

neighbors will not admit more members in the period of routing convergence following a network failure (since AS relationship is static). Hence, for the first type of routing dynamics, we can rule out the possibility that the filter will block a valid IP packet. We illustrate this as follows: consider an IDPF enabled AS v that is on the best route from s to d. Let $u = \mathbf{bestU}(s, d, v)$, and let $U = \mathbf{feasibleU}(s, d, v)$. A link or router failure between u and s can have three outcomes: 1) AS u can still reach AS s, and u is still chosen to be the best upstream neighbor for packet $M(s, d)$, i.e, $u = \mathbf{bestU}(s, d, v)$. In this situation, although u may explore and announce multiple routes to v during the path exploration process [7], the filtering function of v is unaffected. 2) AS u is no longer the best upstream neighbor for packet $M(s, d)$; another feasible upstream neighbor $u' \in U$ can reach AS s and is instead chosen to be the new best upstream neighbor (for $M(s, d)$). Now, both u and u' may explore multiple routes; however, since u' has already announced a route (about s) to v, the IDPF at v can correctly filter (i.e., accept) packet $M(s, d)$ forwarded from u'. 3) No feasible upstream neighbors can reach s. Consequently, AS v will also not be able to reach s, and v will no longer be on the best route between s and d. No new packet $M(s, d)$ should be sent through v.

Yet another concern of routing dynamics relates to how newly connected network (or a network recovered from a fail-down event) will be affected. In general, a network may start sending data immediately following the announcement of a (new) prefix, even before the route has had time to propagate to the rest of the Internet. In the time that it takes for the route to be propagated, some packets (from this prefix) maybe discarded by some IDPFs if the reachability information has not yet propagated to them. However, the mitigating factor here is that in contrast to the long convergence delay that follows failure, reachability for the new prefix will be distributed far more speedily. In general, the time taken for such new prefix information to reach an IDPF is proportional to the shortest AS path between the IDPF and the originator of the prefix and independent of the number of alternate paths between the two. Previous work has established this bound to be $O(L)$, L being the diameter of the AS graph [7, 25]. We believe that in the short timescales we are discussing, it is acceptable for IDPFs to potentially behave incorrectly, i.e. discarding valid packets originated from the new network prefix, before the corresponding BGP announcements reach the IDPFs. Similarly,

during this short time of period, IDPFs may fail to discard spoofed attack packets. However, given that most DDoS attacks require a persistent train of packets to be directed at a victim, we believe the said operation of IDPFs for failing to discard spoofed attack packets during this time period should also be acceptable.

5.3 Impact of Overlapping Prefixes

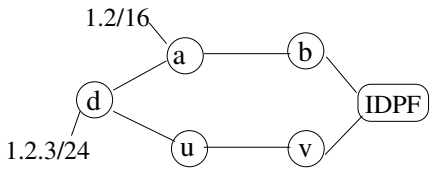


Figure 5: Prefix overlapping and packet filtering.

Ideally, prefixes in an IDPF's filter table should not overlap; any arriving packet can be uniquely matched with a prefix in the table (if a match exists at all). However, due to the almost ubiquitous use of classless addressing, CIDR [17], prefixes in the filter table may overlap. This implies that an incoming prefix may match multiple prefixes in the filtering table. This creates the situation where an attacker in an AS announcing a shorter, less specific, prefix can spoof the IP addresses of an AS with a longer, more specific, prefix (assuming the longer prefix is a subnet of the former).

An obvious way to deal with this is to associate an incoming packet with the *longest* prefix, much in the way that routers forward packets. However, as we show with an example, this leads to incorrect operation. In Fig. 5, AS d is multi-homed to ASes u and a, and it originates the prefix `1.2.3/24`. Provider u propagates this (same) route to v and eventually the IDPF node at the right receives an announcement for `1.2.3/24` (from v) associated with the AS path $\langle v\ u\ d \rangle$. On the other hand, AS a notices that the route announced by d is a subnet from its own larger address space (`1.2/16`). Since reachability for the shorter prefix implies reachability for the longer prefix, AS a simply subsumes the announced prefix into an announcement for its shorter prefix, i.e., `1.2/16`, which eventually reaches the IDPF node (through b) and is associated with AS path $\langle b\ a \rangle$. If the IDPF node uses the longest matching rule to decide between one of the prefixes, *all* packets arriving through b with source address in the longer prefix will be dropped! This is because, from the IDPF's

point of view, it only received a route to the longer prefix from v.

For this reason, the proposed IDPF framework will not conduct longest prefix match to examine if a neighbor is a feasible upstream neighbor for packet $M(s, d)$. Instead, as long as a neighbor has announced the reachability information to a network prefix that contains s, it will be considered as a feasible upstream neighbor. We will further evaluate the impact of overlapping prefixes on the performance of IDPFs in the next section.

5.4 Other Issues

By deploying IDPFs, an AS constrains the set of packets that a neighbor can forward to the AS. Specifically, a neighbor can only successfully forward a packet $M(s, d)$ to the AS after it announces the reachability information of s. All other packets are identified to carry spoofed source addresses and discarded at the border router of the AS. In the worst case, even if only a single AS deploys IDPF and spoofed IP packets can get routed all the way to the AS in question, using an IDPF perimeter makes it likely that spoofed packets will be identified, and blocked, at the perimeter. Clearly, if the AS is well connected, launching a DDoS attack upon the perimeter itself takes a lot more effort than targeting individual hosts and services within the AS. In contrast, ASes that do not deploy IDPF offer relatively little protection to the internal hosts and services. Therefore, an AS has direct benefits to deploy IDPFs. In general, by deploying IDPFs, an AS can also protect other ASes to which the AS transports traffic, in particular, the customer ASes. This can be similarly understood that, an IDPF node limits the set of packets forwarded by a neighbor and destined for a customer of the AS.

The destination address d in a packet $M(s, d)$ plays no role in an IDPF node's filtering decision (*Definition* 5). We make this design decision for the following reasons: 1) We assume that a node u will forward a packet $M(s, d)$ to node v only if $\mathbf{bestR}(v, d)$ has been exported to u by v. 2) By constructing filtering tables based on source address alone (rather than *both* source and destination addresses), per-neighbor space complexity for an IDPF node is reduced from $O(N^2)$ to $O(N)$, where $N = |V|$ is the number of nodes in the graph (the route-based scheme can achieve the same complexity bound [32]).

An IDPF may not be able to catch all spoofed packets forwarded by a neighbor. Note that an IDPF allows all the feasible upstream neighbors for packet $M(s,d)$ to send the packet. However, in reality, exactly one of them will lie on $\mathbf{bestR}(s,d)$ and forward $M(s,d)$. On the other hand, it is worth noting that an attacker in a best upstream neighbor for packet $M(s,d)$ can always spoof the source address s; therefore, route-based packet filters also cannot catch all spoofed packets. In the next section, we will conduct simulation studies to compare the performance of route-based packet filtering with that of the IDPF framework.

6 Performance Studies

In this section we first discuss the objectives of our performance studies and the corresponding performance metrics. We then describe the data sets and specific settings used in the simulation studies. Detailed results obtained from simulations are presented at the end of this section.

6.1 Objectives and Metrics

We evaluate the effectiveness of IDPFs in controlling IP spoofing based DDoS attacks from two complementary perspectives [32]. First, we wish to understand how effective the IDPFs are in *proactively* limiting (if not preventing) the capability of an attacker to spoof addresses of ASes other than his own. Our approach does not provide complete protection and spoofed packets may still be transmitted. Thus the complementary, *reactive* view is also important; we study how the deployed IDPFs can improve IP traceback effectiveness by localizing the actual source of spoofed packets. A third dimension of our simulation studies concerns the issue of incentive, i.e., how an individual AS will benefit from deploying IDPF on its routers.

A family of performance metrics was introduced in [32], and we include them in our own study. Given any pair of ASes, say a and t, $S_{a,t}$ is the set of ASes, from which an attacker in AS a can forge addresses to attack t.[5] For any pair of ASes, s and t, $C_{s,t}$ is the set of ASes, from which attackers can attack t using addresses belonging to s, without such packets being filtered before they reach t.

[5]In other words, if AS $u \in S_{a,t}$, then someone in a can use *any* address in u as the source address in packets sent to t.

To establish a contrast: $S_{a,t}$ quantifies the *pool of IP addresses* that may be forged by an attacker in a to send packets to t without being stopped. On the hand, $C_{s,t}$ is defined from the victim, i.e., AS t's perspective. This quantifies the size of the set of ASes that can forge an address belonging to s in sending packets to t without being discarded along the way. Thus the latter is a measure of the *effort* required, at AS t, to trace the packets to the actual source (there are $|C_{s,t}|$ locations that the packet could have originated from).

6.1.1 Proactive Prevention Metrics

Given the AS graph $G = (V, E)$, we define the *prevention* metric from the point of view of the victim as follows:

$$\phi_1(\tau) = \frac{|\{t : \forall a \in V, |S_{a,t}| \leq \tau\}|}{|V|}$$

$\phi_1(\tau)$, redefined from [32], denotes the proportion of ASes that satisfy the following property: if an arbitrary attacker intends to generate spoofed packets, he can successfully use the IP addresses of at most τ ASes (note that this includes the attacker's own AS). Thus, $\phi_1(\tau)$ represents the effectiveness of IDPFs in *protecting* ASes against spoofing-based DDoS attacks. For instance, $\phi_1(1)$, which should be read as *the fraction of ASes that can be attacked with packets from at most 1 AS*, describes the immunity to *all* spoofing based attacks.[6]

Next, we define a metric from the attacker's perspective. Given $G = (V, E)$, $\phi_2(\tau)$, defined in [32], describes the fraction of ASes from which an attacker can forge addresses belonging to at most τ ASes (including the attacker's own), in attacking any other ASes in the graph.

$$\phi_2(\tau) = \frac{|\{a : \forall t \in V, |S_{a,t}| \leq \tau\}|}{|V|}$$

Intuitively, $\phi_2(\tau)$ is the strength of IDPFs in *limiting* the spoofing capability of an arbitrary attacker. For instance, $\phi_2(1)$ quantifies the fraction of ASes from which an attacker cannot spoof any address other than his own.

[6] As an inter-domain packet filtering framework, IDPFs cannot prevent attackers from forging an address belonging to their own AS. Thus, since $\forall a \in V, a \in S_{a,t}$, we have $\tau \geq 1$.

6.1.2 Reactive IP Traceback Metrics

To evaluate the effectiveness of IDPFs in reducing the IP traceback effort, i.e., the act of determining the true origin of spoofed packets, $\psi_1(\tau)$ is defined in [32], which is the proportion of ASes being attacked that can localize the true origin of an attack packet to be within τ ASes.

$$\psi_1(\tau) = \frac{|\{t : \forall s \in V, |C_{s,t}| \leq \tau\}|}{|V|}$$

For instance, $\psi_1(1)$ is simply the fraction of ASes, which when attacked, can correctly identify the (single) source AS that the spoofed packet was originated from.

6.1.3 Incentives to Deploy IDPF

To formally study the gains that ASes might accrue by deploying IDPFs on their border routers, we introduce a related set of metrics, $\bar{\phi}_1(\tau)$, $\bar{\phi}_2(\tau)$, and $\bar{\psi}_1(\tau)$. Let T denote the set of ASes that support IDPFs.

$$\begin{aligned}
\bar{\phi}_1(\tau) &= \frac{|\{t \in T : \forall a \in V, |S_{a,t}| \leq \tau\}|}{|T|} \\
\bar{\phi}_2(\tau) &= \frac{|\{a \in V : \forall t \in T, |S_{a,t}| \leq \tau\}|}{|V|} \\
\bar{\psi}_1(\tau) &= \frac{|\{t \in T : \forall s \in V, |C_{s,t}| \leq \tau\}|}{|T|}
\end{aligned}$$

Note that these are similar to the metrics defined earlier, i.e., $\phi_1(\tau)$, $\phi_2(\tau)$, and $\psi_1(\tau)$, respectively. However, we restrict the destinations to the set of IDPF enabled ASes, rather than the entire population of ASes.

6.2 Data Sets

In order to evaluate the effectiveness of IDPFs, we construct four AS graphs from the BGP data archived by the Oregon Route Views Project [30]. The first three graphs, denoted G_{2003}, G_{2004}, and

G_{2005} are constructed from single routing table snapshots (taken from the first day in each of the years). While these provide an indication of the evolutionary trends in the growth of the Internet AS graph, they offer only a partial view of the existing connectivity [18]. In order to obtain a more comprehensive picture, similar to [14, 20], we construct G_{2004c} by combining G_{2003} and *an entire year of BGP updates* between G_{2003} and G_{2004}. Note that the Slammer worm attack [28], which caused great churn of the Internet routing system, occurred during this period of time. This had the side effect of exposing many more edges and paths than would be normally visible.[7]

Table 3 summarizes the properties of the four graphs. In the table we enumerate the number of nodes, edges, and AS paths that we could extract from the datasets. We also include the size of the vertex cover for the graph corresponding to individual datasets (the construction is described later). From the table we see that, G_{2004c} has about 22000 more edges compared to G_{2004}, or a 65.9% increase. Also, the number of observed AS paths in G_{2004c} is an order of magnitude more than the observed paths in the G_{2004} data.

Table 3: Graphs used in the performance studies.

Graph	# of Nodes	# of Edges	# of AS paths	VC size (%)
G_{2003}	14516	27406	373350	2124 (14.6%)
G_{2004}	16566	34217	731240	2422 (14.6%)
G_{2005}	18949	39879	811342	2734 (14.4%)
G_{2004c}	18684	56763	7489979	3319 (17.8%)

6.3 Inferring Feasible Upstream Neighbors

In order for each AS to determine the feasible upstream neighbors for packets from source to destination, we also augment each graph with the corresponding AS paths used for constructing the graph [30]. We infer the set of feasible upstream neighbors for a packet at an AS as follows. In general, if we observe an AS path $\langle v_k, v_{k-1}, \ldots, v_0 \rangle$ associated with prefix P, we take this as

[7] Given the lengthy period over which we applied the updates, it is likely that our AS graph includes "stale-edges", i.e., edges that no longer exist. We ignore this effect in our study, noting that AS relationships are quite stable, and thus the number is likely to be very small.

an indication that v_i announced the route for P to v_{i+1}, i.e., $v_i \in \textbf{feasibleU}(P, v_{i+1})$, for $i = 0, 1, \ldots, k-1$.

6.4 Settings of Performance Studies

6.4.1 Routing

Given an AS graph $G = (V, E)$ and a subset of nodes $T \subseteq V$ deploying the IDPFs, the route that a packet takes from source node s to destination node t will determine the IDPFs that the packet will encounter on the way. Consequently, *in order to compute the described performance metrics*, we require the exact routes that will be taken between any pairs of nodes. Unfortunately, there is simply no easy way to get this knowledge accurately. In this paper, as a heuristic, we simply use the shortest path on G. When there are multiple candidates, we arbitrarily select one of them. Note that this knowledge, i.e., the best path from an AS to another, is only required in the simulation studies to determine the IDPFs that a packet may encounter on the way from source to destination. It is *not required in the construction of the IDPFs*. As a consequence, in addition to AS paths, we also include the selected shortest path as a feasible route, if it has not been described in the routing updates observed.

6.4.2 Selecting IDPF Nodes

Given a graph $G = (V, E)$, we select the *filter set*, i.e., nodes in T to support IDPF in one of two ways. The first one, denoted VC, aggressively selects the nodes with the highest degree until nodes in T form a vertex cover of G. In the second method, *Rnd*, we randomly (uniformly) choose the nodes from V until a desirable proportion of nodes are chosen. In the studies that we describe the target proportions are 30% and 50%. The corresponding sets are labeled *Rnd*30 and *Rnd*50, respectively.

6.4.3 BGP Updates vs. Precise Routing

So far we have assumed that the precise global routing information is not available at IDPFs, and they rely on BGP update messages to infer if a packet originated from a prefix can be forwarded by

a specific neighbor. To exactly understand any improvement we gain from *accurate* knowledge of the best route between ASes, we compare performance in two settings. In the first, *BGP updates*, ASes only use the AS paths (as described above) to construct the filters. In the second, *precise routing*, which is identical to the route-based packet filtering in [32], each node in the graph knows the best route for all other pairs and uses that single best route in the filter construction.

6.4.4 Overlapping prefixes

In order to better understand the impact of overlapping prefixes on the performance of IDPFs, we study two different scenarios. In the first scenario, there are no overlapping prefixes announced by *distinct ASes*. In the second case, overlapping prefixes are allowed (or more specifically, we use the real network prefixes announced by each AS in the BGP routing tables and updates). For simplicity, we refer to the former as IDPFs with *non-overlapping prefixes*, and the latter with *overlapping prefixes*.

6.4.5 Network Ingress Filtering

Network ingress filtering [16] is a mechanism that prevents an AS, where the mechanism is deployed, from being used to stage IP spoofing based attacks against host(s) in a different AS. It is reasonable to assume that ASes that deploy IDPFs, being security conscious and network-savvy, will also implement ingress filtering. However, this cannot always taken to be the case, and towards the end of this section, we discuss the case when ingress filtering is not deployed anywhere.

6.5 Results of Performance Studies

The studies are performed with the *Distributed Packet Filtering (dpf)* simulation tool [32]. We extended *dpf* to support our own filter construction based on BGP updates and to deal with overlapping prefixes. Before we describe the simulation results in detail, we briefly summarize the salient findings.

- Although it is difficult to completely protect networks from spoofing-based DDoS attacks (unless filters are near-universally deployed by ASes on the Internet), IDPFs can significantly

limit the spoofing capability of an attacker. For example, with the VC IDPF coverage, an attacker in more than 80% of ASes cannot successfully launch any spoofing-based attack on the Internet (assuming no overlapping prefixes are announced). Moreover, with the same configuration, the AS under attack can localize the true origin of an attack packet to be within 28 ASes, therefore, greatly reducing the effort of IP traceback.

- Overlapping prefixes have a detrimental effect on the performance of IDPFs. However, IDPFs still work reasonably well with overlapping prefixes announced on the Internet. For example, in this case an attacker in about 50% ASes cannot launch any spoofing-based attacks. And for the majority of attack packets, the AS under attack can pinpoint the true origin to be within 79 ASes.

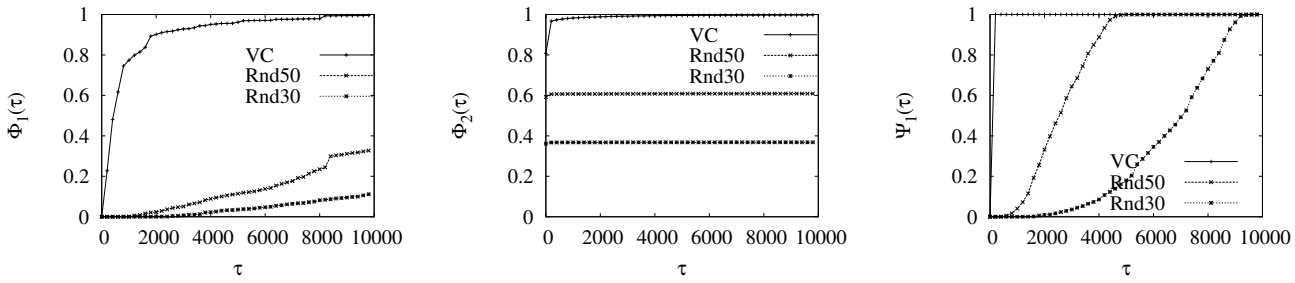
- Network ingress filtering [16] helps improve the performance of IDPFs. However, even without network ingress filtering being deployed in any ASes, an attacker still cannot launch any spoofing-based attacks from within more than 60% of ASes. Moreover, the AS under attack can localize the true origin of an attack packet to be within 87 ASes.

- ASes (and their customers) are better protected by deploying IDPFs compared to the ones that do not. For example, while only about 5% of all nodes on the Internet cannot be attacked by attackers that can spoof IP addresses of more than 6000 nodes, that percentage becomes higher than 11% among the nodes that support IDPFs (with $Rnd30$ IDPF coverage).

6.5.1 IDPFs with BGP Updates and Non-Overlapping Prefixes

To begin with, we study the performance of IDPFs with BGP updates and non-overlapping prefixes. We investigate the impacts of other parameters such as precise routing information and overlapping prefixes in the subsequent sections.

Fig. 6(a) presents the values of $\phi_1(\tau)$ for three different ways of selecting the IDPF node on the G_{2004c} graph: vertex cover (VC) and random covers ($Rnd50$ and $Rnd30$). Note that $\phi_1(\tau)$ indicates the proportion of nodes that may be attacked by an attacker that can spoof the IP addresses of at most τ nodes. In particular, $\phi_1(1)$ is the portion of nodes that are immune to any spoofing-based



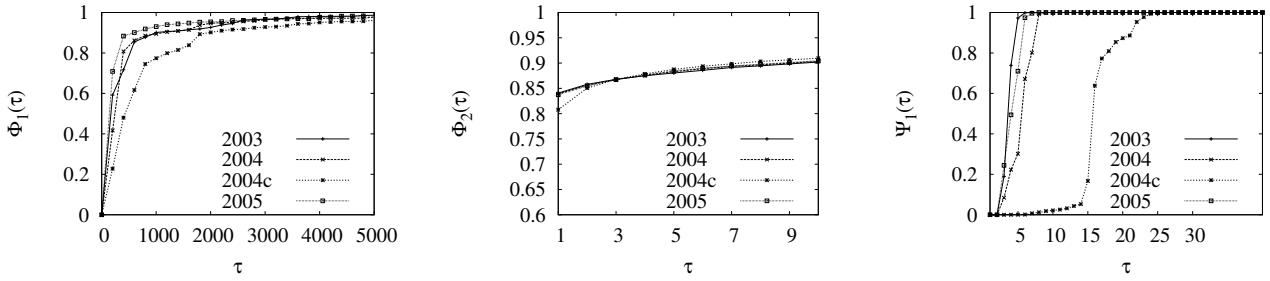
(a) $\phi_1(\tau)$ (b) $\phi_2(\tau)$ (c) $\psi_1(\tau)$

Figure 6: 2004c (With BGP updates and non-overlapping prefixes).

attacks. Unfortunately, it is zero for all three covers. Moreover, as shown in Fig. 8, unless nearly all nodes support IDPFs, we cannot completely *protect* a network from spoofing-based attacks. (This is the case for all the simulations we conducted for this work.) As a consequence, instead of trying to *completely protect* ASes from spoofing-based attacks, we should focus on *limiting* the spoofing capability of attackers, which is indeed feasible as we shall show shortly. The figure also shows that the placement of IDPFs plays a key role in the effectiveness of IDPFs in controlling spoofing-based attacks. For example, with only 17.8% of nodes supporting IDPFs, *VC* outperforms both *Rnd*30 and *Rnd*50, although they recruit a larger number of nodes supporting IDPFs. In general, it is more preferable for nodes with large degrees (such as big ISPs) to deploy IDPFs. Fig. 7(a) shows $\phi_1(\tau)$ for the graphs from 2003 to 2005 (including G_{2004c}). We see that, overall, similar trends hold for all the years examined. However, it is worth noting that G_{2004c} performs worse than G_{2004}. This is because G_{2004c} contains more edges and more AS paths by incorporating one-year BGP updates.

$\phi_2(\tau)$ illustrates how effective IDPFs are in limiting the spoofing capability of attackers. In particular, $\phi_2(1)$ is the proportion of nodes from which an attacker cannot launch any spoofing-based attacks against any other nodes. Fig. 6(b) shows that IDPFs are very effective in this regard. For G_{2004c}, $\phi_2(1) = 0.807857, 0.592325, 0.361539$, for VC, $Rnd50$, and $Rnd30$, respectively. Similar trends hold for all the years examined (Fig. 7(b)).

Recall that $\psi_1(\tau)$ indicates the proportion of nodes that, under attack by packets with a source IP address, can pinpoint the true origin of the packets to be within at most τ nodes. Fig. 6(c) shows



(a) $\phi_1(\tau)$ (b) $\phi_2(\tau)$ (c) $\psi_1(\tau)$

Figure 7: 2003-2005 (VC. With BGP updates and non-overlapping prefixes).

that all nodes can localize the true origin of an arbitrary attack packet to be within a small number of candidate nodes (28 nodes, see Fig. 7(c)) for the VC cover. For the other two, i.e., $Rnd30$ and $Rnd50$, the ability of nodes to pinpoint the true origin is greatly reduced. From Fig. 7(c) we also see that G_{2003}, G_{2004}, and G_{2005} can all pinpoint the true origin of attack packets to be within 10 nodes. However, it is important to note that such graphs are less-complete representations of the Internet topology compared to G_{2004c}.

6.5.2 Impacts of Precise Routing Information

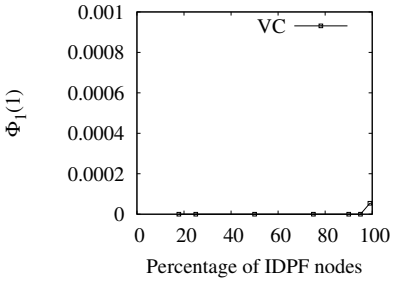


Figure 8: $\phi_1(1)$. The set of IDPF nodes in all cases contains the VC (G_{2004c}).

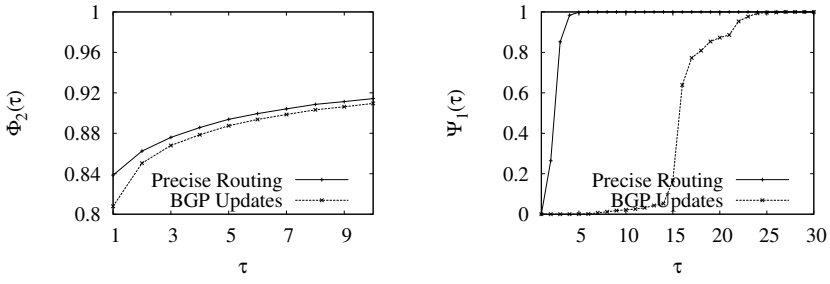
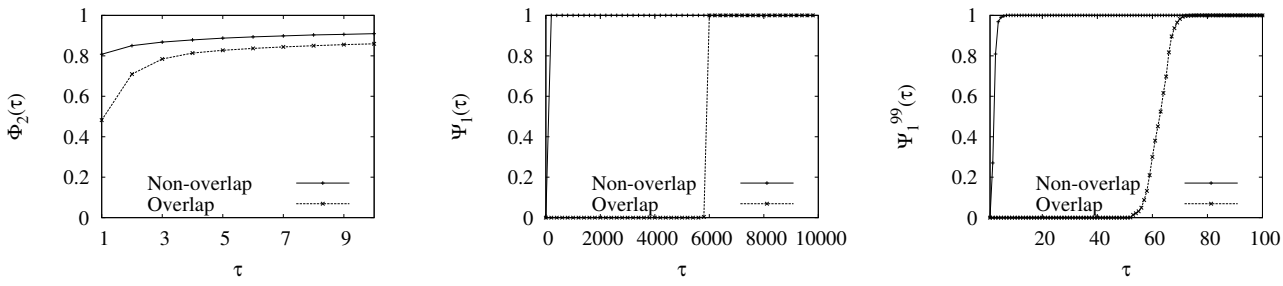


Figure 9: Precise routing information vs. BGP update information (G_{2004c}, VC). Left figure: $\phi_2(\tau)$. Right figure: $\psi_1(\tau)$.

In this section we study the impact of the precise global routing information on the performance

of IDPFs. As shown in Fig. 9, the availability of the precise routing information between any pair of source and destination only slightly improves the performance of IDPFs in comparison to the case where BGP update information is used. For example, while about 84% of nodes cannot be used by attackers to launch any spoofing-based attacks by relying on the precise routing information, there are still about 80% of ASes where an attacker cannot launch any such attacks by solely relying on BGP update information. Similarly, by only relying on BGP update information, an arbitrary AS can still pinpoint the true origin of an attack packet be within 28 ASes, compared to 7 if precise global routing information is available.

6.5.3 Impacts of Overlapping Prefixes



(a) $\phi_2(\tau)$ (b) $\psi_1(\tau)$ (c) $\psi_1^{99}(\tau)$

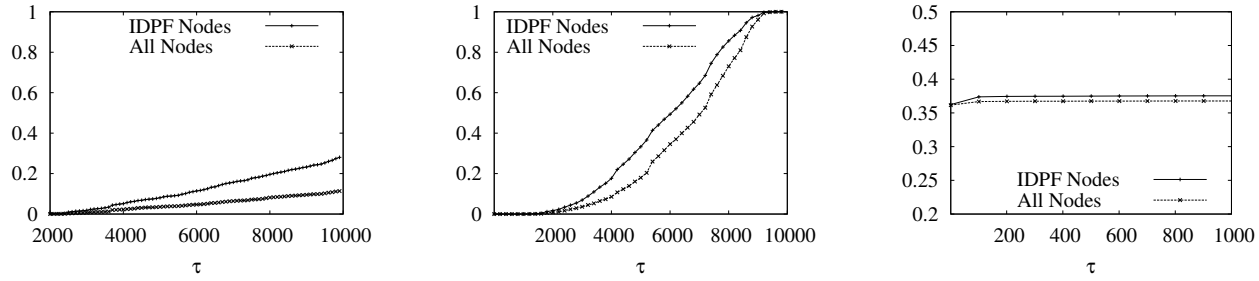
Figure 10: Impacts of overlapping prefixes (G_{2004c},VC).

From Fig. 10(a) we see that overlapping prefixes only have a moderate impact on *limiting* the spoofing capability of attackers. For example, an attacker on about 50% nodes cannot spoof IP addresses of any other nodes. Fig. 10(b) demonstrates that overlapping prefixes may significantly affect the ability of nodes in pinpointing the true origin of an attack packet. However, we speculate that this is caused by ISPs that announce less specific prefixes that contain more specific prefixes announced by other ASes. To verify this, we introduce another metric, $\psi_1^{99}(\tau)$, which is defined with respect to the 99th percentile of $|C_{s,t}|$. Formally,

$$\psi_1^{99}(\tau) = \frac{|\{t : \forall s \in V, P(|C_{s,t}| \leq \tau) = 99\%\}|}{|V|}$$

$\psi_1^{99}(\tau)$ can be interpreted as follows: For an attack packet with an arbitrary IP source address, with 99% probability, we can pinpoint the true origin of the packet to be within τ ASes. Fig. 10(c) presents the values of $\psi_1^{99}(\tau)$. From the figure we see that for more than 99% of IP addresses of attack packets, a node can pinpoint the true origin to be within 79 nodes.

6.5.4 Deployment Incentives



(a) $\bar{\phi}_1(\tau)$ vs. $\phi_1(\tau)$ (b) $\bar{\psi}_1(\tau)$ vs. $\psi_1(\tau)$ (c) $\bar{\phi}_2(\tau)$ vs. $\phi_2(\tau)$

Figure 11: Deployment incentives (G_{2004c}, Rnd30).

One key factor that is responsible for the slow deployment of network ingress filtering is that the deployment of such filtering function directly benefits the rest of the Internet instead of the network that supports it. In contrast, networks supporting IDPFs are better protected than the ones that do not (Fig. 11). In Fig. 11(a) we show the values of $\bar{\phi}_1(\tau)$ (curve marked with *IDPF Nodes*) and $\phi_1(\tau)$ (marked with *All Nodes*). From the figure we see that while only about 5% of all nodes on the Internet cannot be attacked by attackers that can spoof IP addresses of more than 6000 nodes, that percentage increases to higher than 11% among the nodes that support IDPFs. Moreover, as the value of τ increases, the difference between the two enlarges. Similarly, while only about 18% of all nodes on the Internet can pinpoint the true origin of an attack packet to be within 5000 nodes, more than 33% of nodes supporting IDPFs can do so (Fig. 11(b)).

Fig. 11(c) compares the spoofing capability of attackers in attacking a general node on the Internet and that supporting IDPFs. We see that networks supporting IDPFs only gain slightly

in this perspective. This can be understood by noting that, by deploying IDPFs, an AS not only protects itself, but also those to whom the AS transports traffic.

6.5.5 Impacts of Network Ingress Filtering

So far we have assumed that networks supporting IDPFs also employ network ingress packet filtering [16], i.e., attackers cannot launch spoofing-based attacks from within such networks. In this section we examine the implications of this assumption.

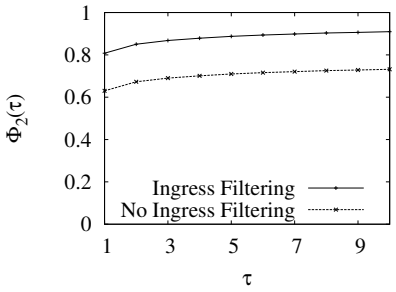


Figure 12: Impacts of ingress filtering (G_{2004c}, VC).

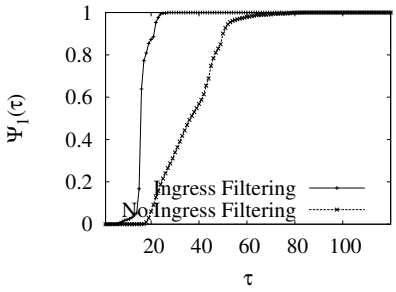


Figure 13: Impacts of ingress filtering (G_{2004c}, VC).

From Fig. 12 we see that ingress packet filtering has only modest impacts on the effectiveness of IDPFs in limiting the spoofing capability of attackers. For example, without network ingress filtering, we still have more than 60% of nodes from which an attacker cannot launch any spoofing-based attacks, compared to 80% when ingress filtering is enabled at nodes supporting IDPFs. As shown in Fig. 13, the impact of network ingress filtering on the effectiveness of IDPFs in terms of reactive IP traceback is also small. Without ingress filtering, an arbitrary node can pinpoint the true origin of an attack packet to be within 87 nodes, compared to 28 when networks supporting IDPFs also employ ingress filtering.

7 Conclusion and Future Work

In this paper we proposed and studied an inter-domain packet filter (IDPF) architecture as an effective countermeasure to the IP spoofing-based DDoS attacks. IDPFs rely on BGP update messages

exchanged on the Internet to infer the validity of source address of a packet forwarded by a neighbor. We showed that IDPFs can be easily deployed on the current BGP-based Internet routing architecture. We studied the conditions under which the IDPF framework can work correctly without discarding any valid packets. Our simulation results showed that, even with partial deployment on the Internet, IDPFs can significantly limit the spoofing capability of attackers; moreover, they also help pinpoint the true origin of an attack packet to be within a small number of candidate networks, therefore, simplifying the reactive IP traceback process. As future work, we plan to investigate how other AS relationship and routing information may help further improve the performance of IDPFs.

Acknowledgment

We thank Kihong Park, Heejo Lee, and Ali Selcuk for providing us with the *dpf* simulation tool, and the Oregon Route Views Project for making BGP routing tables and updates publicly available. The authors are also grateful to Nick Feamster and Jennifer Rexford for the insightful comments on an early version of the paper. Xin Yuan was supported in part by NSF Grants ANI-0106706, CCR-0208892, and CCF-0342540. Jaideep Chandrashekar was supported in part by NSF Grants ITR-0085824 and NSF CNS-0435444, and a Cisco URP grant. Any opinions, findings, and conclusions or recommendations expressed in this paper are those of the authors and do not necessarily reflect the views of National Science Foundation or Cisco Systems.

References

[1] F. Baker. Requirements for ip version 4 routers. RFC 1812, June 1995.

[2] S. Bellovin. ICMP traceback messages. Internet Draft, October 2001. Work in Progress.

[3] R. Beverly. Spoofer project. `http://momo.lcs.mit.edu/spoofer`.

[4] R. Beverly and S. Bauer. The Spoofer Project: Inferring the extent of Internet source address filtering on the internet. In *Proceedings of Usenix Steps to Reducing Unwanted Traffic on the Internet Workshop SRUTI'05*, Cambridge, MA, July 2005.

[5] A. Bremler-Barr and H. Levy. Spoofing prevention method. In *Proc. IEEE INFOCOM*, Miami, FL, March 2005.

[6] CERT. Cert advisory ca-1996-21 TCP SYN flooding and IP spoofing attacks, 1996. `ttp://www.cert.org/advisories/CA-1996-21.`tml.

[7] J. Chandrashekar, Z. Duan, Z.-L. Zhang, and J. Krasky. Limiting path exploration in BGP. In *Proc. IEEE INFOCOM*, Miami, FL, March 2005.

[8] Cisco Systems, Inc. BGP path selection algorithm. `http://www.cisco.com/warp/public/459/25.shtml`.

[9] E. Cooke, F. Jahanian, and D. McPherson. The zombie roundup: Understanding, detecting, and disrupting botnets. In *Proceedings of Usenix Steps to Reducing Unwanted Traffic on the Internet Workshop (SRUTI 2005)*, Cambridge, MA, July 2005.

[10] M. Dalal. Improving TCP's robustness to blind in-window attacks. Internet Draft, May 2005. Work in Progress.

[11] Massive DDoS attack hit DNS root servers. `http://www.internetnews.com/ent-news/article.php/1486981`, October 2002.

[12] Yahoo attributes a lengthy service failure to an attack. `http://www.nytimes.com/library/tech/00/02/biztech/articles/08yahoo.html%`, February 2000.

[13] D. Dean, M. Franklin, and A. Stubblefield. An algebraic approach to IP traceback. *ACM Transactions on Information and System Security*, 5(2):119–137, 2002.

[14] X. Dimitropoulos, D. Krioukov, and G. Riley. Revisiting internet as-level topology discovery. In *Passive and Active Measurement Workshop (PAM)*, Boston, MA, March 2005.

[15] Z. Duan, X. Yuan, and J. Chandrashekar. Constructing inter-domain packet filters to control ip spoofing based on bgp updates. In *Proc. IEEE INFOCOM*, Barcelona, Spain, April 2006.

[16] P. Ferguson and D. Senie. Network ingress filtering: Defeating denial of service attacks which employ ip source address spoofing. RFC 2267, January 1998.

[17] V. Fuller, T. Li, J. Yu, and K. Varadhan. Classless inter-domain routing (CIDR): an address assignment and aggregation strategy. RFC 1519, September 1993.

[18] L. Gao. On inferring autonomous system relationships in the internet. In *Proc. IEEE Global Internet Symposium*, November 2000.

[19] L. Gao and J. Rexford. Stable internet routing without global coordination. *IEEE/ACM Transactions on Networking*, 9(6), December 2001.

[20] R. Govindan and A. Reddy. An analysis of Internet inter-domain topology and route stability. In *INFOCOM (2)*, pages 850–857, 1997.

[21] S. Halabi and D. McPherson. *Internet Routing Architectures*. Cisco Press, 2 edition, 2000.

[22] G. Huston. Interconnection, peering and settlements-part I. *The Internet Protocol Journal*, March 1999.

[23] C. Jin, H. Wang, and K. Shin. Hop-count filtering: an effective defense against spoofed ddos traffic. In *Proceedings of the 10th ACM conference on Computer and communications security*, October 2003.

[24] Srikanth Kandula, Dina Katabi, Matthais Jacob, and Arthur Berger. Botz-4-Sale: Surviving Organized DDoS Attacks that Mimic Flash Crowds. In *Second Symposium on Networked Systems Design and Implementation (NSDI'05)"*, 2005.

[25] C. Labovitz, A. Ahuja, R. Wattenhofer, and V. Srinivasan. The impact of internet policy and topology on delayed routing convergence. In *INFOCOM*, pages 537–546, 2001.

[26] Craig Labovitz, Danny McPherson, and Farnam Jahanian. Infrastructure attack detection and mitigation. SIGCOMM 2005, August 2005. Tutorial.

[27] J. Li, J. Mirkovic, M. Wang, P. Reiher, and L. Zhang. SAVE: source address validity enforcement protocol. In *INFOCOM*, June 2002.

[28] D. Moore, V. Paxson, S. Savage, C. Shannon, S. Staniford, and N. Weaver. Inside the slammer worm. *IEEE Security and Privacy*, 2003.

[29] D. Moore, G. Voelker, and S. Savage. Inferring internet Denial-of-Service activity. In *Proceedings of 10th Usenix Security Symposium*, August 2001.

[30] University of Oregon. Route Views project. http://www.routeviews.org/.

[31] R. Pang, V. Yegneswaran, P. Barford, V. Paxson, and L. Peterson. Characteristics of internet background radiation. In *Proceedings of ACM Internet Measurement Conference*, October 2004.

[32] K. Park and H. Lee. On the effectiveness of route-based packet filtering for distributed DoS attack prevention in power-law internets. In *Proc. ACM SIGCOMM*, San Diego, CA, August 2001.

[33] V. Paxson. An analysis of using reflectors for distributed denial-of-service attacks. *ACM Computer Communications Review (CCR)*, 31(3), July 2001.

[34] Y. Rekhter and T. Li. A border gateway protocol 4 (BGP-4). RFC 1771, March 1995.

[35] S. Savage, D. Wetherall, A. Karlin, and T. Anderson. Practical network support for IP traceback. In *SIGCOMM*, pages 295–306, 2000.

[36] A. Snoeren, C. Partridge, L. Sanchez, C. Jones, F. Tchakountio, S. Kent, and W. Strayer. Hash-based ip traceback. In *Proc. ACM SIGCOMM*, 2001.

[37] J. Stewart. *BGP4: Inter-Domain Routing In the Internet*. Addison-Wesley, 1999.

[38] J. Stewart. DNS cache poisoning - the next generation. Technical report, LURHQ, January 2003.

[39] Team Cymru. The team cymru bogon route server project. `http://www.cymru.com/BGP/bogon-rs.html`.