Constructing Inter-Domain Packet Filters to Control IP Spoofing Based on BGP Updates

Zhenhai Duan
Florida State University
duan@cs.fsu.edu

Xin Yuan
Florida State University
xyuan@cs.fsu.edu

Jaideep Chandrashekar
University of Minnesota
jaideepc@cs.umn.edu

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 address in IP packets. By employing IP spoofing, attackers can evade detection and put the burden on the destination networks, which must take a substantial amount of efforts in order to police the attack packets.**

In this paper we propose an inter-domain packet filter (IDPF) architecture and study its effectiveness in mitigating the level of IP spoofing on the Internet. IDPFs are deployed on border routers of networks and are constructed using the information in BGP updates. A key feature of IDPF is that it does not require global routing information. Based on extensive simulation studies, we show that even with partial deployment on the Internet, IDPFs can not only pro-actively limit the spoofing capability of attackers, but also localize the origin of an attack packet to a small number of candidate networks, which may increase the accuracy of reactive IP traceback schemes.

I. Introduction

Distributed Denial of Service (DDoS) attacks pose an increasingly grave threat to the Internet. This point is driven home on a regular basis by successful DDoS attacks mounted both on popular Internet sites and the infrastructure itself [9], [8]. The existing situation with DDoS attacks is fueled in part by the widespread availability of attack kits and other software, e.g., stacheldracht, Tribal Flood, etc., which allow novice users to launch large scale attacks with relative ease. The complicating factor, which makes policing these attacks hard, is the use of "IP spoofing"—the act of forging the source IP address of an unsuspecting host in the attack packets. By masquerading as a different host, attackers can hide their actual identities and locations, causing significant damage without the fear of being discovered.

In recent times, there have been accounts of attackers utilizing "bot-nets", collections of tens of thousands of compromised hosts (often subverted by worms or viruses) that are used to stage attacks [18]. Since the attacks are being carried out through intermediaries, i.e., the compromised "bots", it is believed that the use of IP spoofing is less of a factor than before. However, recent studies present evidence to the contrary and show that IP spoofing is still a commonly observed phenomenon [23], [25]. In addition, Beverly et. al.[4], [5] describe results showing that large parts of the Internet are still vulnerable to IP spoofing.

IP spoofing remains prevalent due to two significant reasons: first, IP spoofing makes it harder to isolate attack traffic from legitimate traffic, as packets with spoofed source addresses may appear to be from all around the Internet. IP spoofing also presents the attacker with an easy way to insert a level of indirection and shift the burden to the victim being attacked—a substantial amount of effort is required to localize the source of the attack traffic [3], [10], [28], [29]. Second, several attacks use IP spoofing as an integral mechanism, for instance: TCP SYN flood attacks rely on spoofing addresses of hosts which are unable to respond to replies, TCP hijack and man-in-the-middle attacks are carried out by the attacker masquerading as the host at the other end of a valid TCP session [7]; reflector based attacks use IP spoofing to masquerade as some victim hosts that contacts a number of hosts resulting in the victim flooded by replies from all these hosts [27]. These factors lead us to believe that that use of IP spoofing is unlikely to decrease in the near future.

While attackers may insert arbitrary source addresses into the IP packets being sent, they cannot control the actual path taken by these packets to reach the destination. The (inter-domain) path taken (from source to destination) is largely controlled by the Border Gateway Protocol (BGP), which is the de-facto inter-domain routing protocol. 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 BGP, exchanging information about its own networks and others that it can reach (through others). Previously, Park and Lee [26] proposed the route-based packet filters as a way to mitigate IP spoofing. The key intuition in this scheme is that there is exactly a single path between a source and a destination AS (which is the aggregate outcome of the BGP computation of all the nodes along the path). Any packets with addresses from the source AS but forwarded along a different path contain spoofed addresses. While simple, a significant drawback of the scheme is that it requires filter nodes to have knowledge about the routing decisions made by all the ASes in the Internet. Given the distributed, policy-based nature of BGP, this assumption is unrealistic and hence the scheme is impractical in the current Internet routing regime.

In this paper, we build upon the ideas in [26] and propose an Inter-Domain Packet Filter (IDPF) architecture to limit the incidence of IP spoofing on the Internet. We show that the information carried in BGP updates can be used to infer the feasible paths between a source and a destination and that effective IDPFs can be constructed using such information. While IDPF cannot stop all spoofed packets, it reduces the capability of the attackers by limiting the number of IP addresses that an attacker can spoof. In addition, IDPF can localize the origin of an attack packet to a small number of networks, which significantly improves IP traceback accuracy. Hence, IDPF can be a significant step toward eliminating IP spoofing on the Internet.

The key contributions in this paper can be summarized as follows:

- We describe the details of an inter-domain packet filter framework to limit IP spoofing attacks. The IDPFs are constructed by exploiting information that is implicit in BGP route announcements and thus use information that is already available.
- To evaluate the effectiveness of our framework, we conduct extensive simulation studies based on AS topologies extracted from real BGP data provided by the Route-Views project [24]. Our results show that, even with partial deployment, our framework can proactively limit an attacker's ability to spoof packets. When the spoofed packet cannot be stopped, we can localize the attacker to a small number of candidate ASes, thereby reducing the effort and increasing the accuracy of IP traceback schemes.
- A significant shortcoming in many protection schemes is that there is relatively little local benefit to network operators; a global social good does not easily translate to individual networks being benefited. Our results demonstrate that ASes (and their customers) are better protected against spoofing-based DDoS attacks by deploying IDPFs compared to those that do not, which should give sufficient incentives for ISPs to deploy IDPF.

The remainder of the paper is organized as follows. We discuss related work in Section II. Section III provides an abstract model of BGP. In Section IV we present the IDPF architecture. In Section V we discuss practical deployment issues of IDPFs. We perform simulation studies on the effectiveness of IDPFs in Section VI. We conclude the paper and discuss future work in Section VII.

II. Related Work

The idea for IDPF is motivated by previous work carried out by Park and Lee [26], which to our knowledge, was the first effort to evaluating the relationship between topology and packet filtering effectiveness. The authors show that filters deployed in *few* ASes can significantly limit packet spoofing. The main drawback is that the filter nodes require *precise* knowledge of the routing choices made at *all* other ASes. Given the decentralized, autonomous nature of inter-domain routing, this requirement is hard to reconcile. We extend this idea in our own work and demonstrate how filters can be constructed using only *local* routing information.

Unicast reverse path forwarding (uRPF) [2] requires that a packet be forwarded only when the interface that the packet arrives on is exactly the same used by the router to reach the source IP (in 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), described in [17], each *end system* maintains a mapping between IP address aggregates and valid hop counts, i.e., the number of routers on the path from the origin to the end system. Packets that arrive with a different hop count are suspect and are therefore discarded (attackers that are the same distance away from the destination as the spoofed source will still be successful). However, it is not clear how this can deal with attackers who can insert arbitrary hop counts in the spoofed packets. Also, HCF is an end-system based approach; spoofed attack traffic is still delivered to the victim making it ineffective against bandwidth based attacks. In [20], Li et. al., describe 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; the filters can identify and discard spoofed packets. However, a few incremental deployment issues of SAVE remain open [20].

In the Network Ingress Filtering proposal described in [12], 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 Team Cymru Bogon Route Server Project [31] 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. IP packets with source addresses that match the bogon list are filtered out. This mechanism can only filter out packets with unroutable source IP addresses. It has no effects on attack packets carrying routable but spoofed source addresses.

III. Border Gateway Protocol and AS Interconnections

In this section, we describe some of the operation in BGP, focusing on aspects that are essential to this paper (for a complete description, see [30]). We model the AS graph of the Internet as an undirected graph $G = (V, E)$. V is the set of nodes (corresponding to ASes in the Internet), and E is the set of edges between ASes, representing *BGP sessions* between neighboring ASes. To simplify things, we shall assume that there is at most one edge between any pair of nodes (this does not affect the correctness of our scheme and considerably simplifies the description).

Nodes exchange route updates, which may be announcements or withdrawals, with their neighbors to learn of reachability changes to destinations in the graph. Route withdrawals are generated in response to a network being unreachable. In contrast to route announcements, withdrawals only enumerate a set of prefixes and signal that the prefixes are no longer reachable through the neighbor forwarding the withdrawal.

On the other hand, route announcements, generated when an AS learns of a route to some destination, is more expressive. Each route *r* (in a route announcement) describes reachability to a specific prefix *r*.**prefix**. Associated with this prefix are a number of route attributes. Of particular interest is the AS path attribute—the path vector, denoted as *r*.**as_path**, that describes the ordered list of nodes that the route has traversed. Other important attributes include the next-hop, denoted *r*.**next_hop**, which is the IP address of the neighbor that the route was learned from; the local preference, denoted *r*.**local_pref**, which describes the degree of preference of the route (in the AS).

Let $\langle v_k v_{k-1} \dots v_1 v_0 \rangle$ be the AS path associated with route r, i.e., *r*.**as_path**= $\langle v_k v_{k-1} \dots v_1 v_0 \rangle$. This implies that v_0 originates the corresponding network prefix *r*.**prefix**, and also that r has traversed $v_1, v_2, \dots, v_{k-1}$ in that order (before arriving at node v_k). For conciseness, we say that edge $e(v_i, v_{i-1}) \in E$ is on the AS path, or $e(v_i, v_{i-1}) \in r.\textbf{as\_path}$, for $i = k, k-1, \dots, 1$. Importantly, BGP is an incremental routing protocol, i.e., route updates are generated only in response to network events.

To simplify the description, in the rest of this section, we assume that d is the (single) network of interest and all route announcements are about d.

A. Policies and Route Selection

After receiving a route from a neighbor, a node first applies locally defined *import policies* which can either modify specific route attributes or even reject the route outright. Let r be a route received at v from node u. Then **import**$(v \leftarrow u)[\{r\}]$ denotes the (possibly modified) route that has been *transformed* by the import policies. For example, a node can set the value of *r*.**local_pref** to specific values corresponding to the particular neighbors. As another example, an import policy discards the route if it detects a potential routing loop, done by the transform

$$\textbf{import}(v \leftarrow u)[\{r\}] = \{\}, \text{if v} \in \text{r.}\textbf{as\_path}.$$

After the routes are passed through the import policies at node v, they are stored in v's routing table. The set of all routes (learned from different neighbors) is denoted R and represents the set of *feasible* routes that v may use. From among these, v selects a single *best* route to reach the destination (given by the prefix in the route). We can model the best route selection process as **select**[R]. The selection process is essentially a ranking function on the set R, and the route with the highest rank is returned (see [1] for a detailed description of the selection process). For convenience, we represent the output of the selection process as **br**(v, d) which reads *best route to destination d at node v.*

Having selected a particular route from R as the most preferred, v then *exports* the route to its neighbors *after* applying neighbor specific export policies. We denote by **export**$(v \rightarrow u)[\{r\}]$ the route obtained by transforming r, which is to be sent to neighbor u. The export policies determine if a route should be forwarded to the neighbor, and if so, modifies the route attributes as specified by the local *export policies*. For instance, a typical export policy is to set the next hop route attribute, i.e., when v exports route r to its neighbor u, it sets *r*.**next_hop** to v. Yet another export policy—one that is (also) always applied—is to prepend the AS path in the route with v (perhaps multiple times).

B. AS Relationships influencing routing policies

The specific routing policies at a node are largely determined by relationships with its different neighbors. Three relationships are defined between a pair of neighbors in the AS graph: *provider-customer*, *peering*, and *sibling* [16], [14]. In a provider-customer relationship, the customer network pays the provider to receive transit service so that it can reach the rest of the Internet. In a peering relationship, which is settlement free, each of the nodes sends the other the traffic originating in its own network, as well as its customers. However, they do not transit traffic from other peers or customers to each other. Hence one of them cannot use the other to reach the rest of the Internet (specifically the parts not owned by the peer or any of its customers). In a sibling relationship, two ASes provide mutual transit service to each other. Note that two sibling nodes can be regarded as the provider (and customer) of one another.

The three different commercial relationships directly influence the export policies that are applied to routes being sent to neighbors. This follows because announcing a route to a neighbor is an implicit guarantee that the traffic (for the destination in the route) will be forwarded by the announcing node. Since these policies play an important role in our scheme we enumerate the specific export policies for each type of commercial relationship that a node may have with its neighbor.

r1. **Exporting to a provider:** The AS will announce routes to its own networks and also routes to its customers. It will *not* announce routes learned from other providers and peers.

r2. **Exporting to a customer:** The AS will announce *all* routes that it knows about. These include routes for its own networks and those learned from customers, peers and providers.

r3. **Exporting to a peer:** The AS will announce routes for its own networks, along with routes to its customers. It will *not* announce routes learned from other providers and other peers.

r4. **Exporting to a sibling:** The AS will announce routes to all the routes that it knows about. These includes routes to its networks, routes learned from customers, providers and peers.

As we will discuss in the next section, the enumerated export policies are not *complete*. In a few cases, ASes may choose to apply less restrictive policies to satisfy traffic engineering goals. However for the moment, to simplify the description of the IDPF framework, we shall take the **r1-r4** as

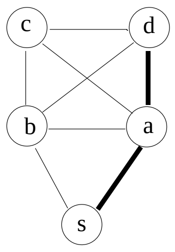


Fig. 1. Route-based packet filtering.

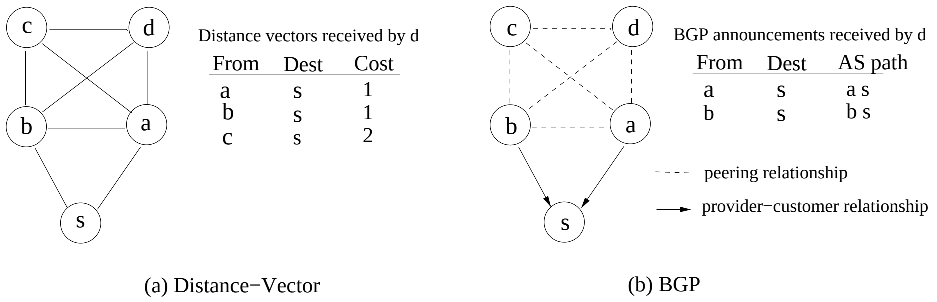


Fig. 2. What route updates reveal.

the working set of export policies. Subsequently, towards the end of the next section, we discuss the more general case.

IV. Inter Domain Packet Filters

In this section we describe the intuition behind the Inter-Domain Packet Filter (IDPF) architecture and how we exploit information contained in route updates to construct an IDPF. In addition, we also formally describe the IDPFs and establish their correctness. Then we discuss the case where ASes have routing policies that are not compatible with IDPFs.

A. Motivating IDPFs

Let us denote by $M(s,d)$ a packet (we also abuse notation and sometimes use the same notation to describe a *family* of packets) whose source address is s (or more generally, the address belongs to AS s), and destination IP address d.

Now a valid M(s,d), i.e., where the source address has not been spoofed, should only be seen by ASes in **br**(s,d).**as_path** as it goes from source to destination. The essential idea in *route-based packet filtering* [26] is exactly this; $M(s,d)$ can only be forwarded by the nodes on the path **br**(s,d).**as_path** and nodes not on this path can consider the packet to be invalid. This can be stated precisely as follows:

Definition 1 (Route-Based Packet Filtering): Node u can forward a packet $M(s,d)$ to node v, if and only if $e(u,v) \in \mathbf{br}(s,d).\mathbf{as\_path}$. Otherwise, the source address of the packet must have been spoofed, and the packet will be discarded at node v.

Fig. 1 illustrates the intuition in route-based packet filtering. Here, a packet $M(s,d)$ can only be forwarded along the best route between node s and d, that is, the route $\langle s\ a\ d\rangle$, shown by the bold line. Thus, if node a were to receive $M(s,d)$ from b, it should be discarded since a knows that $b \notin \mathbf{br}(s,d).\mathbf{as\_path}$.

In BGP, route selection is a local decision, i.e., s computes **br**(s,d) based on the set of routes in its routing table and preferences that an operator in AS s has defined. This information is not available at any other nodes in the network. The implication is that there is no way for an arbitrary node $a(\neq s)$ to know exactly what nodes lie on **br**(s,d).**as_path**. Consequently, route-based packet filtering is mainly of theoretical interest and cannot be deployed in a realistic setting on the Internet.

To address this shortcoming, we propose an inter-domain packet filter (IDPF) architecture that does not require global knowledge of a node's best route selection. Instead, the IDPF deployed in node v uses only the route updates exchanged by v and its *immediate* neighbors to construct the filters. To illustrate the intuition behind IDPF, we use an example to contrast the route propagation behavior in traditional Distance Vector (DV) protocols with that in policy based protocols such as BGP.

In traditional DV protocols, e.g., RIP, a node will receive reachability information (along with a cost estimate) to *all* destinations from *each* of its neighbors. In other words, a node will always export a route to its neighbor (if it knows of one). However, the nature of relationships between ASes mandates a certain amount of information-hiding when ASes exchange routes using BGP (as described in **r1-4**). Consider the examples in Fig. 2. To simplify the description we focus the reachability to s as d sees it. Nodes in Figure 2(a) run a distance vector protocol (such as RIP), while all the nodes in Fig. 2(b) run BGP as described in the previous section.

In Fig. 2(a), the distance vectors received by node d are shown alongside (the costs are hop counts). An important observation is that d learns a route to s from *each* of its neighbors. In this specific example, d would select the route from a since it is the lowest cost route. Contrast this situation with that in Fig. 2(b). Here, we have that nodes a, b, c, and d have mutual peering relationships, and that a and b are providers to s. Here, since a and b are providers to s, they will both export a route for s to d. However, node c, which also receives the route from a and b will not forward this to d (recall rule **r3**: *a node will not export routes learned from a peer to a different peer*). On the other hand, node c will also not export to a and b a route to d (recall again rule **r3**). As a consequence, node c can never be on the best route, $\mathbf{br}(s,d)$, between s and d. Therefore, c should not be forwarding any packets $M(s,d)$ to d.

Thus, an AS should not expect to see IP packets from a neighbor unless it previously received a route (to the source in the packet) *from the same neighbor*. This requires further explanation. When u exports a route to v (say to destination s), the implication is that v and its downstream neighbors can send traffic to s using u. However, when we bring in **r1-4**, the following is implied: if u exports a route to v, then the policies also permit v to export a route (say to destination d) to u. By this, v becomes a valid forwarder of traffic from u

(and transitively from s) destined to d. Note that this does not make any claim if this edge will actually be used, but simply that it is a valid edge to carry traffic for the pair s and d.

Of course, this intuition is not of much use as a filtering scheme if *all* of a node's neighbors announce routes to s. Fortunately, this is rarely the case. It turns out that even if there are a large number of possible paths between a source and destination, due to ASes using **r1-4** when exporting routes, only some of those paths are valid to carry traffic from the source (to the destination). If we can enumerate these valid paths, then we can filter out packets that are forwarded on the invalid paths. These must necessarily contain spoofed addresses, as otherwise, they would have been forwarded on one of the valid paths. Notice that the size of the set (of valid paths) determines how effective the filtering is. The smaller the set the better the filtering performance. In route-based filtering, the size of the set is exactly 1 and hence this will perform the best. Unfortunately, in BGP, there is no way to distribute this set to all the nodes. In our scheme, we identify the set of *feasible* neighbors at each node that *can* forward packets $M(s, d)$ to the node, based on locally exchanged BGP updates between the node and its immediate neighbors.

B. Constructing IDPFs

In this section, we formalize the notion of IDPFs and discuss its correctness. First we define some terms that will be used later.

Definition 2 (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 propagated by any node.

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

Definition 4 (Best Forwarder): For any node v, neighboring node u is the *best forwarder* for $M(s, d)$ if and only if $e(u, v) \in \mathbf{br}(s, d).\mathbf{as\_path}$. We denote this as **bestF**(s,d,v) $= u$.

Note that the best forwarder is defined from the viewpoint of *packet forwarding*. In light of *Definition* 4, we can restate route-based packet filtering as: *node v will only accept packets of type $M(s, d)$ from the neighbor that corresponds to* ***bestF****(s,d,v)* .

Definition 5 (Feasible Forwarder): For node v, node u is a *feasible forwarder* for packets $M(s, d)$, if

$$\mathbf{export}(u \rightarrow v)[\{\mathbf{br}(u, s)\}] \neq \{\}$$

Further, for node v, we denote the set of all feasible forwarders for $M(s, d)$ as **feasibleF**(s,v). Note that this set is allowed to be empty.

In contrast to the previous definition, feasible forwarders are defined from the viewpoint of *BGP route propagation*. And it should also be clear that a feasible forwarder may not be the best forwarder. Below we establish that, however, the best forwarder must be a feasible forwarder.

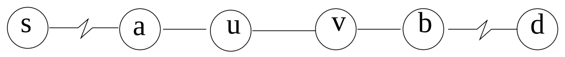


Fig. 3. The best route from AS s to AS d

Lemma 1: For any source s and destination d, and a node v on the best route from s to d, we have, $\mathbf{bestF}(s, d, v) \in \mathbf{feasibleF}(s, v)$, if the routing system is stable.

We use the following notation in the proof: if node b is a provider of node a, and node c is a provider of node b, we call c an *indirect* provider of a.

Proof: Consider the best path from s to d, which is known only at s. Without lose of generality, assume that this path traverses the edge (u, v), as depicted in Fig. 3. Clearly, a route to d was exported to b (perhaps traversing other nodes between b and d); otherwise b would not be on the best path from s to d. Furthermore, we have a route propagation sequence from b to v, then u, going all the way to s. This chain exists by the construction of **br**(s,d). Also, we can verify that u is either an (indirect) provider of s or an (indirect) provider of d (or provider of both). Otherwise, u must have violated one of the exporting policies **r1-4** by announcing to node a a route to node d. In the following we assume that u is an (indirect) provider of s. We can also establish the lemma, in a similar manner, where u is either a provider of d or a provider to both s and d. We omit them here to avoid repetition.

With this observation, node u announces to customer a the best route to node d, independent of the relationship between u and v. Moreover, given that u is an (indirect) provider of s, u announces to v the route to node s, based on the exporting policies **r1-4**, no matter what relationship nodes u and v have. Therefore, node u is a feasible carrier of packets originated from node s, and this proves the lemma. ■

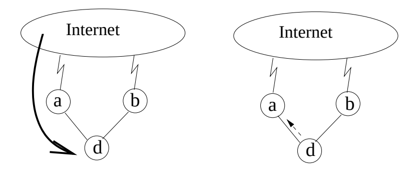
The filters are defined at a node specific to each neighbor. At node v, we can define the filter for node u as follows:

Definition 6 (Inter-Domain Packet Filtering (IDPF)): Node v will accept packets $M(s, d)$ forwarded by a neighbor node u, if and only if $\mathbf{export}(u \rightarrow v)[\{\mathbf{br}(u, s)\}] \neq \{\}$[1]. Otherwise, the source address of the packet must have been spoofed, and the packet should be discarded by node v.

Theorem 1: An IDPF as defined in *Definition* 6 is correct.

Proof: Without loss of generality, consider source s, destination d, and a node $v \in \mathbf{br}(s, d).\mathbf{as\_path}$ such that v deploys an IDPF filter. Let $u = \mathbf{bestF}(s, d, v)$. From Lemma 1, u is also a feasible forwarder, and $\mathbf{export}(u \rightarrow v)[\{\mathbf{br}(u, s)\}] \neq \{\}$ from *Definition* 5. From *Definition* 6, packets $M(s, d)$ forwarded from u to v will not be filtered out by v. ■

[1]As a technical detail, the condition should be $\mathbf{import}(v \leftarrow u)[\mathbf{export}(u \rightarrow v)[\{\mathbf{br}(u, s)\}] \neq \{\}$. That is, not only is $\mathbf{br}(u, s)$ is exported to v by u, but also accepted by v. However, for clarity of our presentation, we ignore the effects of import policies in the paper.



(a) Route preference (b) Conditional advertisement

Fig. 4. Traffic engineering for multi-homing ASes.

C. Routing Policy Complications

As bandwidth costs decrease, there is an increasing trend for stub ASes[2] to be multi-homed, i.e., obtain connectivity via more than one service provider, to improve overall reliability of their Internet connectivity. In some cases, it is desirable for a multi-homed AS to use one provider as the primary or preferred provider, and treat others as back-up providers. This is illustrated in Fig 4(a). Here, ASes a and b are the providers for AS d. Among these, d wishes to use a as the primary provider and use b as a backup provider (primary connectivity is marked with the bold-line). There are several different mechanisms to achieve this goal.

A particular way to achieve this is by using so-called *BGP Conditional Advertisement* or *Selective Announcements*, as depicted in Fig. 4(b). Here, AS d announces its routes to provider a (but not to b), causing all incoming traffic to arrive via a. If the primary route through a fails, then AS d will send a route to provider b and this causes subsequent inbound traffic to arrive via b. When the original primary link is restored, d withdraws its route from b, causing traffic to switch back to arriving via a. Here, note that d does *not* announce its route(s) to b as long as the link to a is available. However, it may still use b to send outgoing traffic. If, in this situation, b deploys an IDPF, traffic from d will be blocked.

It is important to note that *selective* announcement is very different from the case where a route is prohibited by **r1-4**. In the present case, d *chooses* not to export the route even though it is allowed to (within the guidelines defined in **r1-4**). Obviously, this behavior cannot be supported within the IDPF framework. To be consistent with IDPF, the AS can choose an alternate method to achieve the same goal, perhaps using AS path prepending (to make the route through b less preferred) or using scope limiting community attributes. A stronger case can be made for this by noting that the mechanism of selective announcements have undesirable side effects [32].

In spite of all this, if for some reason, an AS u still chooses to employ selective announcements, i.e., it *chooses* not to announce a route to a neighbor (even though it can under **r1-4**), we suggest the following rule to be applied at u:

r5. Restricted conditional advertisement policy: If an AS can announce a route (originated by a specific network) to a neighbor, but chooses not to do so, then the AS must not forward any traffic from the network to the particular neighbor.

[2]ASes that have no customers and do not provide transit service for others.

If each AS on the Internet follows the routing policies **r1-5**, we can establish the correctness of IDPFs as defined in *Definition* 6 on the Internet. The proof is similar to the one of Lemma 1 and Theorem 1 and we omit it here.

Before we move onto the next section, where we discuss the practical deployment issues of IDPFs, we briefly describe a few properties. For conciseness, we describe a node that implements IDPF as an *IDPF node*.

First, by deploying IDPFs, an AS constrains the set of packets $M(s,d)$ that a neighbor can forward to the AS. Specifically, a neighbor can only successfully forward to the AS the packets $M(s,d)$ to whose source addresses the neighbor has announced the reachability information. All other packets are identified to carry spoofed source addresses and discarded at the border router of the AS. 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 transport 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 to a customer of the AS.

Secondly, for node u to forward a packet $M(s,d)$ to an IDPF node v, we only require that a route *r*.**prefix**=s be exported by v (to u). This does not restrict v's options to reach s. In fact, v might learn of paths to s from a number of other neighbors and might choose one of them (rather than u) as the best route to reach s. Then, possibly, the path from s to d does not match the reverse path. Thus, the IDPF architecture that we describe makes no assumption about the symmetry of routes, which is a problem in some other filtering schemes [2].

Thirdly, the destination address d in a packet $M(s,d)$ plays no role in an IDPF node's filtering decision (*Definition* 6). This requirement is imposed by the following reasons: 1) In a path-vector routing protocol such as BGP, a node v does not know if it is on the best route of a packet $M(s,d)$. So destination addresses do not provide additional information for filtering packets; 2) We assume that a node u will forward a packet $M(s,d)$ to node v only if **br**(v,d) has been exported to u by v. 3) By constructing filtering tables based on source address alone (rather than *both* source and destination address), 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 [26]).

Lastly, an IDPF may not be able to catch all spoofed packets forwarded by a neighbor. Note that, in contrast to the route-based packet filters, an IDPF maintains a *set* of feasible forwarders of $M(s,d)$. They are allowed to send packets $M(s,d)$. However, in reality, exactly one of them will lie on **br**(s,d) and forward $M(s,d)$. On the other hand, it is worth noting that an attacker in a best forwarder of packets $M(s,d)$ can always spoof the address s, therefore, route-based packet filters also cannot catch all spoofed packets.

V. Practical Deployment Issues of IDPFs

In this section, we discuss some aspects of the IDPFs that relate to its implementation and deployment in the real Internet.

A. Incremental Deployment

From the description in Section IV, it should be clear that the IDPFs can be deployed independently in each AS. As discussed previously, there are strong advantages to enabling IDPF in an AS: even though 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.

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". The filters that we describe in this paper are *domain level filters*. Thus an IDPF node is required to track the specific prefixes announced by each neighbor. Typically, we expect that BGP speaking routers will also support IDPF, however this is not a strict requirement. In the case that an IDPF node is *not* a BGP router, it needs to obtain the corresponding prefix announcement information from the existing BGP speaking routers.[3]

In the normal operation for an IDPF node, when a packet arrives at the ingress of the network, it needs to be associated with the specific neighbor that forwarded the packet. This association is trivial if the ASes connect via dedicated circuits.[4] Subsequently, the IDPF matches the address against the set of prefixes announced by the specific neighbor. If a matching prefix exists, then the packet is forwarded to the immediate destination in the AS or further routed towards the final destination. Otherwise, it is dropped at the ingress of the network at the IDPF node.

B. Routing Dynamics on IDPF performance

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 periods of convergence, i.e., as the routers are computing alternate routes, packet forwarding may be affected—packets may be dropped or delayed. While we are not really affected one way or the other by packet losses (if spoofed IP packets are dropped, well, thats a good thing!). However, we wish to ascertain if IDPFs can behave incorrectly, i.e., drop packets that are in fact valid, when the network is in flux.

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

[4]However, this is slightly more involved if the ASes connect at a *shared* interconnect (such as an Internet eXchange Point), which involve packets switched over a common backplane. The easiest way to perform the association is by using the link layer headers in the forwarded packet.

To address this concern, 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 forwarders will not admit more members in this period (since the route export policies are static). Hence, we can rule out the possibility that the filter will block a valid IP packet and we illustrate this with an example: consider an IDPF enabled AS v that is on the best route from s to d. Let $u = \mathbf{bestF}(s, d, v)$, and let $U = \mathbf{feasibleF}(s, v)$. 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 forwarder of packets $M(s, d)$, i.e, $u = \mathbf{bestF}(s, d, v)$. In this situation, although u may explore and announce multiple routes to v during the path exploration process, the filtering function of v is unaffected. 2) AS u is no longer the best forwarder[5] of packets $M(s, d)$; another feasible forwarder $u' \in U$ can reach AS s and is instead chosen to be the new carrier. 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) packets $M(s, d)$ forwarded from u'. 3) No feasible forwarders can reach s. In this case, AS v will also not be able to reach s. As a consequence, v will no longer be on the best route between s and d, and no new packets $M(s, d)$ should be sent through v.

Yet another concern, relating to routing dynamics, relates to how newly created network 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 filtered 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 [6], [19]. Note that most DDoS attacks require a persistent train of packets to be directed at a victim, a process that takes a certain amount of time to ramp up, we believe that in the short timescales we are discussing, the said operation of the IDPFs is quite acceptable (for failing to filter out attack packets). Similarly, it should be acceptable for IDPFs to potentially behave incorrectly (i.e. filtering out valid packets originated from the new network prefix) within the discussed short timescales.

C. Impact of Overlapping Prefixes

Ideally, prefixes in an IDPF's filter table should not overlap. Then, any arriving packet can be uniquely matched with a

[5]Possibly because it cannot reach s or perhaps another *more preferred* neighbor becomes available.

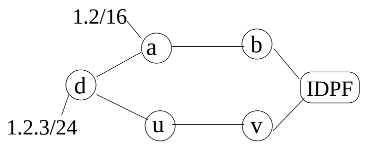


Fig. 5. Prefix overlapping and packet filtering.

prefix in the table (if a match exists at all). However, due to the almost ubiquitous use of classless addressing, CIDR [13], prefixes in the filter table may overlap. This implies that an incoming prefix may not match a distinct prefix in the filter 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 of course that 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 the figure 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 the 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. We will further evaluate the impact of overlapping prefixes in the next section.

VI. 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 specific settings used in the simulation studies. Detailed results obtained from simulations are presented at the end of this section.

A. Objectives and Metrics

We evaluate the effectiveness of IDPFs from two complementary perspectives [26]. From the point of view of a proactive approach, we wish to understand how effective the IDPFs are in limiting (if not preventing) the capability of an attacker to spoof addresses from 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 that are not filtered before the reach the destination. A third dimension addresses the issue of incentive, i.e., why, and how, an individual AS will benefit from deploying IDPF on its routers.

A family of performance metrics was introduced in [26], 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,[6] For any pair of ASes, s and t, $C_{s,t}$ is the set of ASes, attackers from which can attack t using addresses belonging to s, without such packets being filtered before they reach 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 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 checked 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).

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 [26], 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 attackers 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.[7]

Next, we define a metric from the attacker's perspective. Given $G = (V, E)$, $\phi_2(\tau)$, defined in [26], 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.

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, we define $\psi_1(\tau)$, which is the proportion of ASes being attacked

[6]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.

[7]As described earlier, we 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 $\tau \geq 1$.

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.

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 $\bar{\psi}_1(\tau)$.

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

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

B. 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 [24]. The first three graphs, denoted G_{2003}, G_{2004}, and G_{2005} are constructed from single routing table snapshops (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. In order to obtain a more comprehensive picture, similar to [11], [15], 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 [22], 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.[8]

Table I summarizes the properties of the four graphs. In the table we enumerate the number of nodes, number of edges and the number of 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.

[8]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.

TABLE I
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%)

C. Settings of Performance Studies

1) Routing: Given an AS graph $G = (V, E)$ and a subset of nodes $T \subseteq V$ deploy 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, to compute the described performance metrics, we require the exact routes that will be taken between any pairs of nodes. Unfortunately, and this goes back to the shortcomings in route-based packet filtering, 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 and *not in the construction of the IDPFs*. In general, IDPFs simply use route announcements sent by neighbors. In the simulations however, we also include the shortest path selected as a valid path, since it might not be described in the routing updates observed.

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 we describe the target proportions are 30% and 50%. The corresponding sets are labelled $Rnd30$ and $Rnd50$, respectively.

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. This information is inferred from observed AS paths in routing updates (and/or routing table snapshots) corresponding to the different datasets. 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 an indication that v_i announced the route for P to v_{i+1}, for $i = 0, 1, \ldots, k-1$.

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) to construct the filters. In the second, *precise routing*, which is exactly the study in [26], each node in the graph knows the best route for all other pairs and uses that single best route in the filter construction.

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*.

5) Network Ingress Filtering: Network ingress filtering [12] is a mechanism that prevents an AS, where there the mechanism is deployed, from being used to stage attacks against host(s) in a different AS. It is reasonable to assume that ASes that deploy IDPF, 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.

D. Results of Performance Studies

The studies are performed with the *Distributed Packet Filtering (dpf)* simulation tool [26]. We extended *dpf* to support our own filter construction, i.e., using BGP updates, and also to deal with overlapping prefixes. Before we describe the simulation results in detail, we briefly enumerate the salient findings.

- Although 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 VC IDPF coverage, an attacker cannot successfully launch a spoofing attack using addresses from at least 80% of ASes 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 modest impact on the performance of IDPFs. For example, even if overlapping prefixes are announced on the Internet, 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.
- Similarly, network ingress filtering [12] also slightly degrades IDPF performance. 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 (using Rnd30 IDPF coverage).

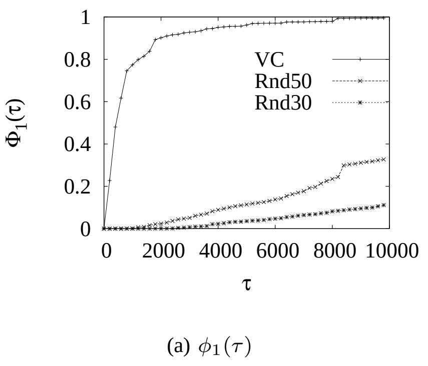
1) IDPFs with BGP Updates and Non-Overlapping Prefixes: To begin with, we study the performance of IDPFs with BGP updates and non-overlapping prefixes. We investigate the impacts of other parameters such as precise routing information and overlapping prefixes in the subsequent sections.

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 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* an network from spoofing-based attacks. (This is the case for all the simulations we conducted for this work.) As a consequence, 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 $Rnd30$ and $Rnd50$, although they recruit a larger number of nodes supporting IDPFs. 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.

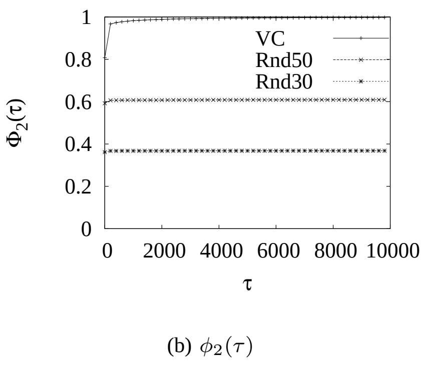
$\phi_2(\tau)$ illustrates how effective IDPFs are in limiting the spoofing capability of attackers. In particular, $\phi_2(1)$ is the proportion of nodes from which an attacker cannot launch any spoofing-based attacks against any other nodes. Fig. 6(b) shows that IDPFs are very effective in this regard. 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 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 can also see that G_{2003}, G_{2004}, and G_{2005} can all pinpoint the true origin of an attack packet 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}.

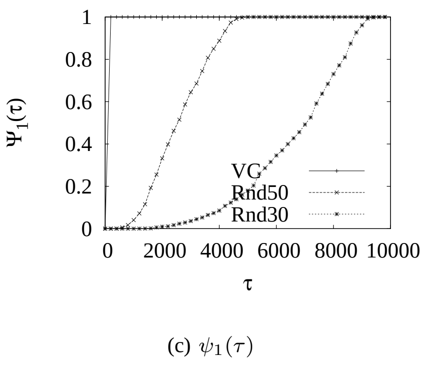
2) Impacts of Precise Routing Information: In this section we study the impact of the precise global routing information on the performance of IDPFs. As shown in Fig. 9, the availability of the precise routing information between any pair of source and destination only slightly improves the performance 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



(a) $\phi_1(\tau)$

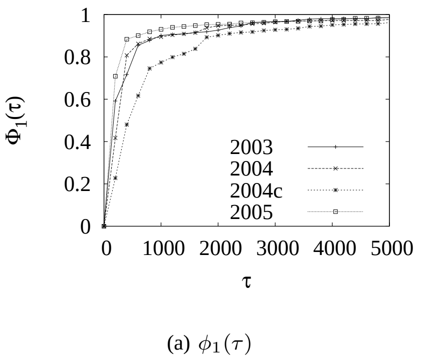


(b) $\phi_2(\tau)$

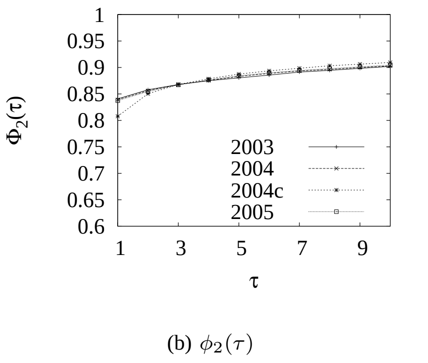


(c) $\psi_1(\tau)$

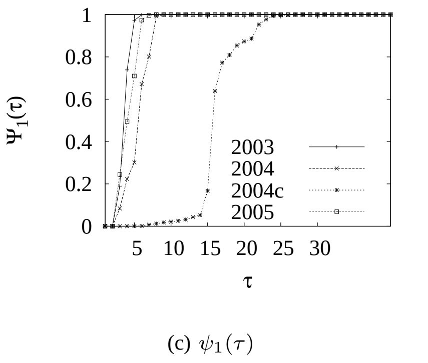
Fig. 6. 2004c (With BGP updates and non-overlapping prefixes).



(a) $\phi_1(\tau)$



(b) $\phi_2(\tau)$



(c) $\psi_1(\tau)$

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

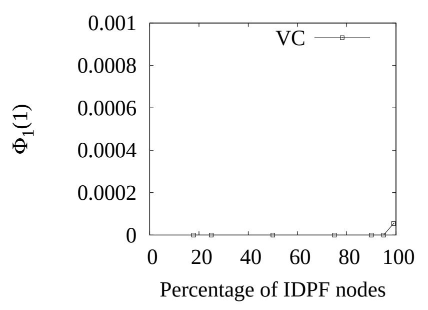
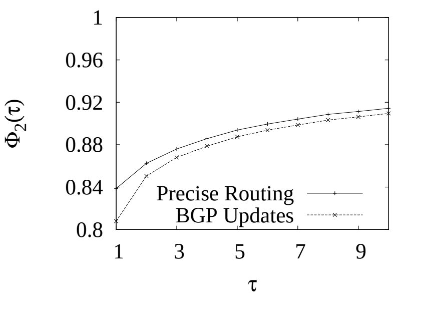


Fig. 8. $\phi_1(1)$. The set of IDPF nodes in all cases contains the VC (G_{2004c}).



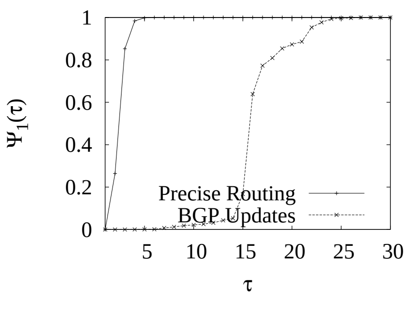
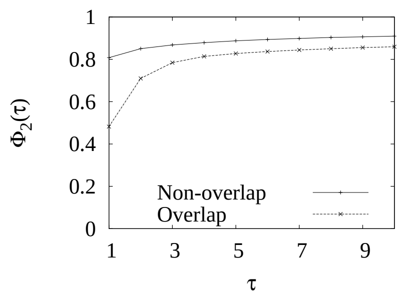


Fig. 9. Precise routing information vs. BGP update information (G_{2004c}, VC). Left figure: $\psi_1(\tau)$. Right figure: $\phi_2(\tau)$.

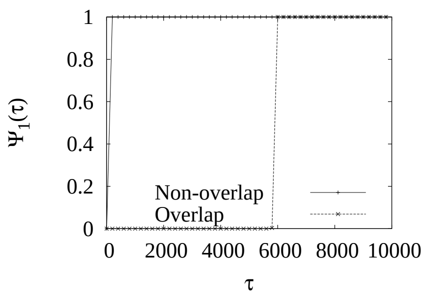
attacks by replying on the precise routing information, there are still about 80% of ASes where an attacker cannot launch any such attacks by solely relying on BGP update information. Similarly, by only relying on BGP update information, an arbitrary AS can still pinpoint the true origin of an attack packet be within 28 ASes, compared to 7 if precise global routing information is available.

3) Impacts of Overlapping Prefixes: 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,

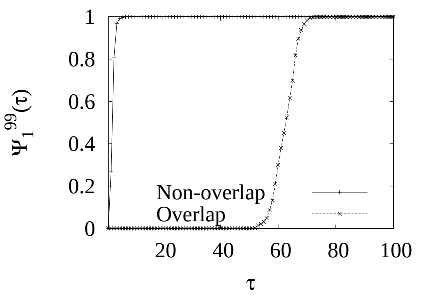
$$\psi_1^{99}(\tau) = \frac{|\{t : \forall a \in V, P(|C_{s,t}| \leq \tau) = 99\%\}|}{|V|}$$



(a) $\phi_2(\tau)$

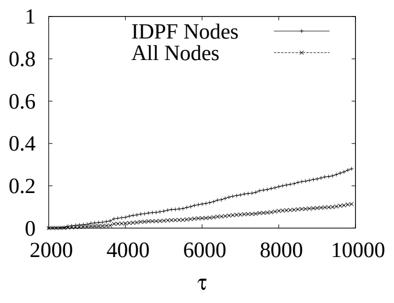


(b) $\psi_1(\tau)$

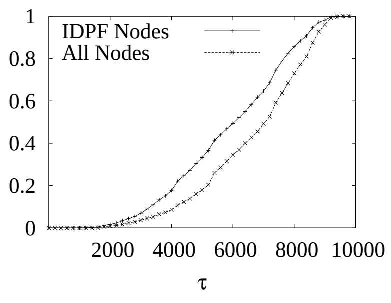


(c) $\psi_1^{99}(\tau)$

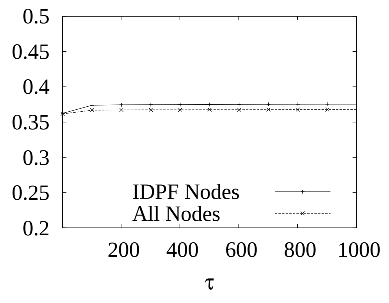
Fig. 10. Impacts of overlapping prefixes (G_{2004c},VC).



(a) $\bar{\phi}_1(\tau)$ vs. $\phi_1(\tau)$



(b) $\bar{\psi}_1(\tau)$ vs. $\psi_1(\tau)$



(c) $\bar{\phi}_2(\tau)$ vs. $\phi_2(\tau)$

Fig. 11. G_{2004c}, Rnd30.

$\psi_1^{99}(\tau)$ 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.

4) Deployment Incentives: 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.

5) Impacts of Network Ingress Filtering: So far we have assumed that networks supporting IDPFs also employ network ingress packet filtering [12], i.e., attackers cannot launch spoofing-based attacks from within such networks. In this section we examine the implications of this assumption.

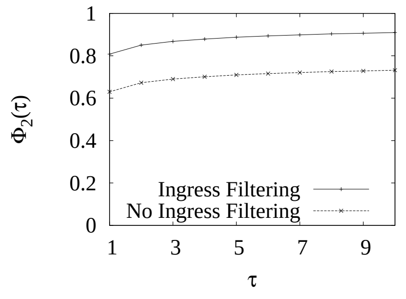


Fig. 12. Impacts of ingress filtering (G_{2004c}, V).

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.

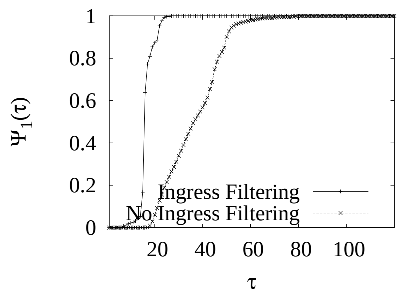


Fig. 13. Impacts of ingress filtering (G_{2004c}, VC).

VII. 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. Our simulation results showed that, even if partially deployed 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 study the cost introduced by the filtering function on the forwarding path of packets. We also plan to investigate schemes to improve the performance of IDPFs [21].

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.

References

[1] BGP path selection algorithm. `http://www.cisco.com/warp/public/459/25.shtml`.

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

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

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

[5] 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.

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

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

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

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

[10] 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.

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

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

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

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

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

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

[17] 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.

[18] 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.

[19] 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.

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

[21] Z. Mao, L. Qiu, J. Wang, and Y. Zhang. On AS-level path inference. In *Proc. ACM SIGMETRICS*, Alberta, Canada, June 2005.

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

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

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

[25] 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.

[26] 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.

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

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

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

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

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

[32] F. Wang and L. Gao. The impact of routing protocol and policies and Internet resilience. In *Proc. ACM SIGCOMM*, September 2004. Poster.