Filtering Source-Spoofed IP Traffic Using Feasible Path Reverse Path Forwarding with SDN

Kevin Benton*, L. Jean Camp, Tim Kelley, Martin Swany
School of Informatics and Computing, Indiana University, Bloomington, IN, USA.

* Corresponding author. Email: KTBenton@Indiana.edu
Manuscript submitted July 24, 2015; accepted April 23, 2016.
doi: 10.17706/ijcce.2016.5.6.441-454

Abstract: Source IP address spoofing is still a significant problem on today's Internet. Recent DDoS attacks, which combined source IP spoofing and amplifying UDP services, have resulted in attack traffic volumes exceeding hundreds of gigabits per second. In this work we argue that the ingress packet filtering solutions proposed in BCP 38 more than 13 years ago have failed to solve the issue due to fundamental incentive misalignment. We present an SDN implementation of feasible path reverse path forwarding which tier 2 ISPs could implement using OpenFlow switches at peering points with no impact to the performance of their routers. We show how an SDN solution can handle error cases more gracefully than current reverse path forwarding implementations. We illustrate that this proposal is well-aligned with the economic incentives of the adopting parties and furthermore does not require ubiquitous adoption to create network-wide immunity. We describe our open code implementation on OpenFlow. Finally, we discuss the limitations of this filtering approach.

Key words: IP Spoofing, SDN, Distributed denial of service attacks, Internet routing.

1. Introduction

Amplification attacks are an endemic challenge in the network. Solutions proposed in BCP 38 more than a decade ago have failed to meet the challenge, in no small part because of incentive misalignment and second order economic effects. Specifically, an ISP that pays to adopt BCP 38 receives no direct benefit, any benefit goes to the intended targets of DDoS attacks. The secondary concern is that a very high percentage, arguably approaching universal adoption, is necessary for there to be significant benefit to the solution proposed in BCP 38.

For a solution to be viable, it has to meet non-technical requirements. It has to provide value to the party that invests in the solution, not someone up or downstream. In addition to providing immediate benefit, it needs to be low cost. In addition, in addition to having low initial cost, the certainty of that cost over time must be bounded.

Amplification attacks continue because current solutions require that nearly every network provider has the incentive, capacity, and expertise to prevent source IP spoofing. Amplification attacks are an economic and technical network problem. The incentive analysis and incentive aware design warrants revisiting this old problem with a solution that provides a clearer path to adoption.

Today the economics of networking are being changed by SDN[1] Feasible Path Reverse Path Forwarding (FPRPF) becomes low cost on SDN so that implementation becomes reasonably affordable in operational terms. FPRPF becomes feasible in fact as well as name.

In addition to changing the economics of networking, we show that SDN can also change the economics of defending against amplification attacks. Because of the vaunted SDN control/data plane separation, IP filtering can be inherently simplified and made affordable by eliminating the need for specialized routers with the capacity to filter high-bandwidth connections. Filtering can be conducted using relatively low-cost switches without impact to the existing routing infrastructure so there is limited implementation risk.

Source IP address spoofing is a simple attack. Attackers replace the source IP field in the packets they are sending onto the network with addresses that differ from the one assigned to them. The chosen address to replace the original source address depends on the type of attack; however, in all cases, source spoofing makes it very difficult for the receiver of the malicious traffic to make any assertions about the true source. It is up to transport layer protocols (e.g. TCP, UDP) to provide this source verification if necessary.

One early spoofing attack exploited the *rlogin* protocol [1] by setting the source IP of the malicious *rlogin* packets to a known trusted host by the target and guessing the TCP sequence numbers, which were predictable at the time [2]. IP spoofing attacks on TCP-based protocols were largely eliminated by making TCP sequence numbers infeasible for an attacker to predict [3].

Since UDP has no implicit protection against these attacks, protecting against spoofing is left up to each individual protocol that leverages UDP. In 2008, Dan Kaminsky revealed a major spoofing vulnerability in most DNS implementations that allowed an attacker to preempt a response to a recursive name server by spoofing the source address of the authoritative server it was attempting to query [4]. In a similar manner to TCP, the workaround for this particular vulnerability was randomization (source port in this case) to make a successful attack highly improbable.

Other than these few higher-level vulnerabilities, source IP spoofing has been primarily used for denial-of-service attacks. In a DoS or DDoS, the attacker has a strong motivation to keep the true source of the packets hidden from the target of the attack. First, by constantly changing the source IP address, it makes it very difficult for the target to filter out the attack by source. Second, if the targets don't know the real source of the attack, they can't send abuse notifications to the service providers of the attack nodes to have them taken down. While these attacks have always been powerful, an attacker typically required a large botnet to generate a large enough volume of traffic to take down a moderately trafficked website. However, amplification attacks have made IP spoofing attacks very powerful for even small botnet operators [5].

In the section immediately following, we provide an overview of source IP spoofing and background on the amplification attacks that have made IP spoofing problematic for service operators in recent years. In Section 3, we cover the current filtering approaches and introduce our approach. In Section 4, we describe the requirements for an incentive-aligned design, as grounded in the economics of security to motivate the need for a new approach. In Section 5, we discuss our prototype implementation of this filtering solution that leverages an OpenFlow switch to avoid a performance impact on ISP core routers. In Section 6, we discuss the limitations of this approach and our planned future work to quantify these limitations. Finally, we conclude in Section 7.

2. Amplification Attacks

In a basic spoofing DDoS attack, the attacker creates packets with the destination field set to the target's

[1]Software-Defined Networking: Essentially allows remote, fine-grained control of data-plane forwarding decisions on network hardware.

IP address and the source field set to an address the target is unlikely to block (e.g. a major website). The size of the attack in this case is limited by the aggregate upstream bandwidth of the attack nodes, which can be limiting for attackers using a botnet comprised of nodes on residential Internet connections with limited upstream bandwidth.

In an amplification attack, the attacker creates packets with the source field set to the target's IP address and the destination field set to the IP address of a public server that answers queries for a given protocol. The attacker populates the body of the packet with a query that is expected to generate a large response from the query server, and sends it onto the network. When the spoofed packet arrives at the query server, the server generates a response and sends it to the target's address (the source field of the query packet). Fig. 1 illustrates the mechanics of an amplification attack.

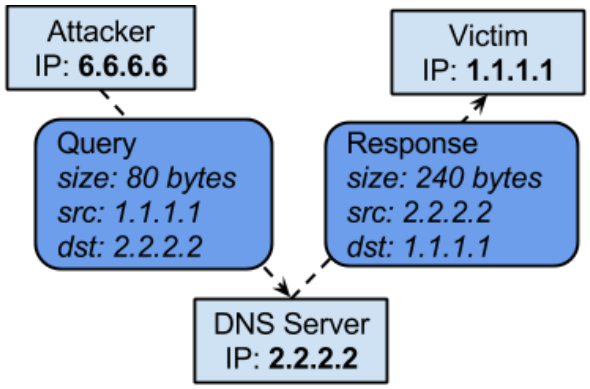


Fig. 1. Traffic flow of an amplification attack with a 3x amplification factor.

The attacker can repeat this process using thousands of different servers simultaneously to prevent the target and the DNS servers from rate limiting based on IP addresses. Therefore the attacker gains the multiplication factor provided by the intermediary servers while the target remains effectively as defenseless as it would be to a regular spoofing attack.

For a successful amplification attack, an attacker must find an unauthenticated query that result in a larger response than the query itself. There are many UDP protocols that exhibit this property, with some providing an amplification of over 4000 times the size of the original query [5]. Additionally, both NTP and DNS exhibit this property to varying degrees and they are widely deployed, giving attackers a deep pool of query servers to utilize as their attack amplifiers. A recent NTP amplification attack has led to the largest DDoS attack ever recorded, with a total traffic volume of 400 gigabits per second [6].

To address this problem, there have been some proponents of removing all of the services that respond in an amplifying fashion to anonymous queries. *CloudFlare*, a company that provides DDoS mitigation via their own content distribution network, has referred to open DNS resolvers as "the scourge of the Internet" due to the role they are playing in amplification attacks [7] and is advocating for the majority of open DNS resolvers to be shut down. However, we believe this approach is flawed for a few reasons. First and foremost, as illustrated by the 400 Gbps NTP attack and the research by Rossow in [5], DNS is not the only effective amplification protocol. Second, open resolvers are providing a useful service when a local resolver is unreliable/untrusted. Third, authoritative and root name servers must always answer queries for the entire Internet; so, even if all of the open resolvers are eliminated, DNS could still be used as an amplification protocol (especially as the prevalence of signed DNSSEC responses increases). Finally, shutting down open resolvers is fundamentally treating a symptom rather than the true cause of the problem, which is IP spoofing. If IP spoofing can be significantly mitigated, it will solve the amplification attack problem without eliminating public UDP services.

3. Ingress Filtering

The simple technical solution to IP spoofing is for every ISP to drop traffic on its edge interfaces[2] if it is sourced from addresses outside of the subnets directly attached to those interfaces. This approach was published in a "best current practices" document in May of 2000 as BCP 38 [8] and was revised with guidelines for multihomed networks as BCP 84 [9] in March of 2004.

There are three approaches to implementing spoofing filtering outlined in the BCP filtering documents, which are defined as modes of unicast reverse path forwarding (referred to as *RPF* for the rest of this paper). The three approaches are "strict mode", "loose mode" and "feasible mode". Each one has slightly different criteria for determining if the source IP address on a received packet is allowed on that interface.

3.1. Strict Mode

Strict mode states that a router should drop packets received on an interface if the router would not use that same interface to send traffic to the source IP address. This mode breaks any networks that leverage asymmetric routing.

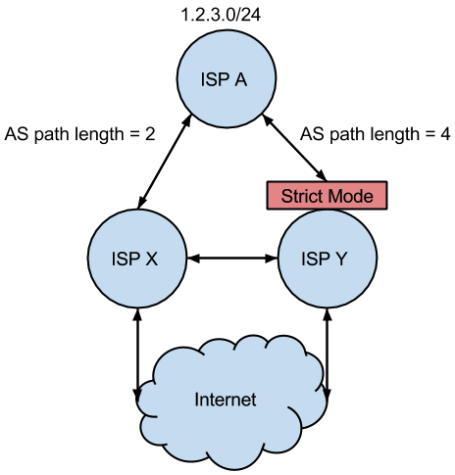


Fig. 2. Strict mode uRPF breaking case.

Take Fig. 2 for example; assume **ISP A** purchases primary connectivity from **ISP X** and a secondary connection for high priority traffic from ISP Y. To prevent traffic from flowing into their network over the high priority link, **ISP A** extends the path length for its BGP advertisements to **ISP Y**. **ISP Y** will believe that the shortest route to the **ISP A** network (1.2.3.0/24) is via **ISP X**. If **ISP Y** implemented strict mode RPF on the interface to **ISP A**, it will then drop any packets from **ISP A** sourced from the 1.2.3.0/24 network, which makes the link unusable by **ISP A**.

3.2. Loose Mode

Loose mode states that a router must contain some path to the source IP address in a received packet, otherwise it should drop it. While this mode doesn't break the connectivity in the scenario above, it's so lenient that it's almost completely ineffective on routers at Internet peering points. This is because these routers normally have a copy of the global routing table, which contains a path to any advertised address space on the Internet. Therefore, the only dropped packets would be ones with source addresses belonging to networks that don't appear in the global routing table. While this is somewhat helpful, this does nothing to address amplification attacks, which use the target of the attack as the source of the spoofed packet.

[2]Interfaces that do not connect to other ISPs.

3.3. Feasible Mode

The last mode is feasible mode, in which the router verifies that the interface on which a packet is received *could* be used to send traffic to the source IP of the packet, even if it's not currently the preferred route. In Fig. 2, if **ISP Y** used feasible mode instead of strict mode and received traffic from **ISP A** sourced from the 1.2.3.0/24 network, it would recognize that it's a valid path and permit the traffic because of the BGP advertisement it received on that link, even though it was a higher cost path.

Feasible mode achieves the goals for our use case. As long as ISPs always advertise connectivity to network addresses from which they are sourcing traffic, upstream ISPs can safely perform feasible path RPF on their peering links. We discuss the limitations of this assumption more in section VI, but for now we assume that this is a reasonable assumption for Tier 3 ISPs since they usually don't provide transit for other ISPs without advertising connectivity to those ISPs.

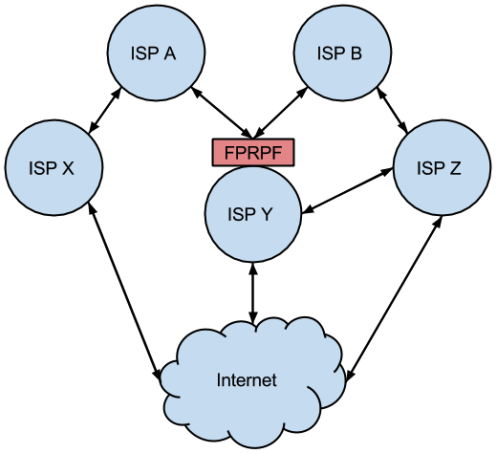


Fig. 3. Feasible path reverse path forwarding topology.

Fig. 3 illustrates our target feasible path RPF implementation by **ISP Y**. **ISP A** and **ISP B** are Tier3 ISPs. **ISP X**, **ISP Y** and **ISP Z** are Tier 2 ISPs. When **ISP Y** implements feasible path RPF on its Tier 3 facing peering interface, it doesn't impact any legitimate traffic sourced from **ISP A** or **ISP B**, regardless of the current preferred route to either ISP.

Unfortunately, feasible mode RPF comes at a cost. It requires the router to keep its entire routing information base (RIB)[3] in expensive memory fast enough to be referenced for every packet. This is in contrast to normal forwarding where the router only keeps its forwarding information base (FIB)[4] in this high-speed memory. Because this memory is expensive, vendors want to put as little of it in their hardware as possible, which frequently ends up ruling out feasible path RPF.

For example, Cisco devices do not have a feasible path RPF implementation. Instead, they allow an access-control list to be matched when a packet fails strict mode RPF [10]. Therefore, to implement feasible path RPF on a Cisco router, each RPF interface would require an ACL containing all of the advertised routes received on that interface, which could require thousands of entries.

3.4. Limited Adoption of Edge Filtering

With more than 10 years since these practices were published, IP spoofing is still a significant problem today. We believe there are two main reasons why this continues to be a problem.

First, the primary suggestion is to only implement it on interfaces that don't peer with other ISPs, which

[3]The RIB keeps track of all possible paths to networks that peers have advertised.
[4]The FIB contains the currently selected routes to each network.

eliminates any 'defense in depth'. Therefore, essentially 100% adoption is required across all ISPs to be effective. This is due the asymmetric advantage granted to attackers by amplification attacks. Based on the 4000 times multiplier [5], a single ISP failing to filter a compromised host with a 100 Mbit uplink could result in attack volumes of 400 Gbps, which is about 10% of the peak volume observed daily at the highest volume Internet exchange point in the world (Deutscher Commercial Internet Exchange)[5].

Second, there is a lack of economic incentive for edge ISPs to filter out this kind of traffic. There are no direct consequences for allowing spoofed traffic because the source in these attacks is, by definition, unknown. Additionally, the majority of the traffic in most consumer-facing ISPs is downstream traffic flowing towards the customers, so filtering upstream traffic does little to alleviate congestion and imbalances on their networks. With no direct upside and the potential downside of implementation costs, including possible support costs due to outages caused by incorrect filtering, it makes little economic sense for an edge provider to implement this filtering. This conclusion is found in other work examining other security filtering performed by ISPs (e.g. IDSs, malware scanning, spam filtering, etc.) [11].

To address these issues, our SDN solution implements feasible path RPF between Tier 2 and Tier 3 ISPs. The use of SDN eliminates any extra load on the ISP routers and offers more flexibility with regards to handling spoofed traffic (e.g. send to deep packet inspection instead of immediately dropping). The placement between Tier 2 and Tier 3 ISPs is based on an incentive-aligned design discussed in detail in the following section.

Our implementation runs on OpenFlow switches at peering points with a low implementation cost, no impact to the performance of the routers, and near zero marginal cost of operation. We identify the limitations of this reverse path filtering and show how an SDN solution can handle these error cases more gracefully than current reverse path forwarding implementations. Our prototype is implemented as a modified POX controller [12] under an Apache license as described in Section 5.

We now discuss the economics of this proposal and the incentive-aligned approach that makes reconsideration of filtering for DDoS emerge again as an interesting topic. SDN is changing the economics of networks. We leverage that change using economics of security.

4. Economics of Security

Security economics has long addressed incentive-aware design and the particular difficulties of incentive-aware design in security. Formal application of the economics of security began with the acknowledgement that incentive-aware design was and is needed for security technologies to be adopted [13]. Following that, Camp and Wolfram identified the existence of externalities in security choices as a difficulty in incentive-aware design [14]. Externalities are neither good nor bad, as both positive and negative externalities exist. Externalities are simply the effect on a firm's or a person's decision that has consequences for others and for which there is no compensation (e.g., not reflected in the price of the good). Varian illustrated that due to the public goods nature of some system-wide security phenomena, even with incentive alignment, investment in security would be less than optimal [15]. Later Herley summarized these points in [16].

Many people are familiar with economics of security from the attempt to price spam through Dwork and Naor's proof of work (PoW) proposal for email [17]. PoW failed to work for several reasons, each of which is an economic phenomena described in more detail below. PoW works only when every- one uses it. This requirement for universal adoption is called a *coordination effect*. Second, the early adopters bear high costs (paying to send) that benefit only the recipient. Thus early adopters have to show a high degree of *altruism*. If a cost high enough to harm spammers would cripple legitimate but intense users of email is an open

[5]3996 Gbps was the peak taken from their stats website on May 3rd, 2015.

question in the literature [18], [19]. This argument is based on the observation that spammers face a very different cost of producing email, as operators of botnets do not pay for the hardware or operating costs of the machines they hijack [20], i.e., a failure to recognize different *production frontiers*.

Choices of technologies are often interdependent in that one entities' choice depends on the choices of others. In economics, *network goods* are goods in which the value of my choosing to use (or not use) a product is a function of how many other people use the product. The value for one user is a function of scope of use by all users. The result of that change in value is referred to as *network effects* or *network externalities*. There are network externalities in many goods that no computer scientist or engineer would consider related to a network. For example, something as simple as popularity or size of a fandom[6] can make associated goods more readily available, e.g. *Lord of the Rings* shirts and fan fiction are now much more widely available than before the movie-length fan fiction of Peter Jackson.

Incentive-aligned design has been used widely, if not consistently, in the intervening decades. Yet even when such systems are proposed, the focus has been on first order incentive effects [21]-[23] rather than second-order system effects. That is, the designs are focused on the immediate costs and benefits to the provider, and fail to address second order requirements. Network externalities, of which coordination effects are a classic example, are an important part of incentive alignment.

Because we discuss different procession frontiers and allocation of benefits, we need to distinguish different actors in the network. To do this we use a standard classification of autonomous systems with three categories: Tier 1, 2, and 3 networks. To avoid any ambiguity, we define these as follows:

- Tier 3 — ISPs that provide ''last-mile' Internet connectivity to companies and consumers.
- Tier 2 — ISPs that charge money to Tier 3 and other Tier 2 ISPs to provide connectivity to the wider Internet.
- Tier 1 — Large Internet backbone ISPs that peer with other Tier 1 ISPs for free and charge Tier 2 and larger Tier 3 ISPs for peering.

In the following sections we present the requirements for a design that is incentive-aligned.

4.1. Incentive-Aligned Design Questions

In this paper we argue that we are implementing incentive-aligned design. Above we introduced some basic concepts in economics of security. Here we answer the basic questions of economics.

4.1.1. What is the problem?

What functionality is needed to defeat a class of attack? System design in any case must begin with a clear security goal that addresses the basic points of leverage for classes of attacks, not individual methods within a class. This is a fundamental condition for a successful system design, as thousands of known security threats exist and new threats are reported daily.

A defense with a larger scope and scale will be more valuable than a narrower targeted defense, due to greater benefits. Attackers are not passive in the face of defender innovations. Thus creating a defense for a class of attacks (i.e. IP spoofing) rather than a particular attack (e.g., open DNS resolvers) makes it more difficult for attackers to continue the same class of assaults with a minor change in their approach. The ability to defend against a larger category of attacks is a result of a defense that has economies of scope.

A security technology may have positive network externalities. Thus when considering DDoS, the proposed technology ideally decreases the overall efficacy of the attack as well as providing immediate benefit to the adopter. If the effect of a technology is simply to shift the attack to others or even make other targets more attractive, it can still be incentive-aligned on the first order. While it is incentive-aligned it has negative network externalities; leaving one party better off at the cost of others. Thus, the goal of our design

[6]A subculture composed of fans of a common interest.

is to harden the network as a whole. When a botnet is taken down, everyone on the network experiences fewer attacks as the resources are removed from the attackers. The primary or first order benefit is to the people who no longer have machines which are contaminated by malware; however, the are larger scale second order effects as many people experience a marginal decrease in malicious activities from those no-longer-subverted machines. When a machine ceases to send spam, there are positive secondary effects for everyone but the spammer and those who purchase spamming services.

4.1.2. Who benefits?

The benefits must also be clear to the adopter. Altruism is not a basis for predicting diffusion of a technology.

Simply because one party can prevent a risk does not mean it wants to do so. The goal here is to identify the parties that are structurally capable of and interested in ending DDoS. Obviously, there are many Tier 3 ISPs who want to end DDoS, yet the ability of a single Tier 3 ISP to do so is quite limited. Considering second order benefits enables identification of optional adoption targets.

When a product must interoperate with another product, such compatibility creates second order effects. This is most clear in gaming systems. The gaming platform that has the most desirable games is the most desirable platform. Networking standards clearly have high levels of second order effects. Thus, any design should ideally not only offer immediate benefit to the adopting party but should also create an overall increase in network security. The proposed technology prevents a Tier 2 from an internal attack and prevents the cost of sending significant data as a source of DDoS attacks. The benefit increases as more ISPs adopt the technology. A secondary advantage of this approach is that the contributions of each ISP to DDoS will become more visible, as the amplification origination is increasingly flagged. Incentives to provide support for malicious actions will decrease when such actions are more transparent.

4.1.3. Who pays?

Early adopters may inherently bear costs for others, particularly learning and training costs. Solutions that are beyond the ability of an organization or individual to adopt will not be adopted. For example, consider route leaks. In theory, route leaks are easy to avoid. Avoidance requires correct configuration and thus such events should be rare, from a naive incentive-alignment perspective. In practice, these happen on a regular basis. The requirement for consistent, correct router configuration is not aligned with the abilities of those who adopt routers.

Similarly, complex network management approaches or real time responses that require high levels of expertise cannot be targeted at small network operators. Thus the IPv4 filtering example presented here targets Tier 2 networks as having both the incentive and the ability. Proposals that require changes in business practice, complete workforce retraining, or other unrealistic actions will not be adopted; nor will technologies that require a leap in organizational competence or commitment to security.

Recall that second order effects apply to choices in networking for three reasons. Network technologies are interdependent, path dependent, and require compatibility. These influence costs as well as benefits.

For example, a security technology may have positive or negative network externalities. A negative network externality would occur if the attacker is seeking a least-effort path. Creating an attack that simply shifts the attacks does not serve the network as a whole. For example, if it is easiest to attack a web server that uses Apache or IIS, then having one server switch from the low security platform causes a negative externality for the remaining users of the low security platform assuming the same number of attackers and attacks. Critics have argued that the purchase of vulnerabilities creates incentives to search for vulnerabilities and also provides protection only to subscribers of the purchasing organization. Were this true (the authors assert no opinion on this matter) then the negative externalities would include increased discovery of vulnerabilities and increased targeting of non-subscribing parties. Of course, it is reasonable to

create technologies that have negative network externalities, yet as part of incentive alignment our proposal does not do so.

4.1.4. What works?

Incentive-aligned design provides guidance on fundamental characteristics of any solution. The technology must not only work; it must be demonstrably working. Benefit should be immediate and clear.

Costs reductions or the potential for cost avoidance should be clear. With this proposal a single Tier 2 network can prevent its customers from being a source of flooding. If an amplification attack occurs within one network, the provider has a very strong incentive to identify and block the sender. If it occurs in two networks, one of these networks must implement filtering to prevent the attack from working. On average an amplification packet will cross at least two Tier 2 networks[7] to fulfill its amplification task.

No feasible solution requires universal adoption or coordinated shift in technologies. Any solution that requires an SDN network without BGP is not feasible at least at this time.

Solutions should be targeted at the entities with both the incentive and the ability to change the overall equilibrium of the network. The Tier 2 networks have both the incentive and ability to adopt the proposed technology. As the scope of the attack increases, the likelihood that such a packet encounters a filtering Tier 2 increases[8]. Tier 2 networks leveraging SDN are the point where incentives meet abilities and costs align with benefits.

5. Implementing RPF in an OpenFlow Switch

There have been other projects examining the use of SDN at BGP exchanges [24], [25]. However, these have been focused on replacing the entire BGP process with an SDN solution. We believe our work is the first that leverages SDN to augment the exchange point with extra functionality without replacing the existing BGP routers.

To implement the feasible path filtering outside of the router, we required a way to programmatically install ACLs into a filtering device based on the BGP updates received by that router. The flexibility afforded by the OpenFlow [26] protocol enabled us to do this in a way that wasn't tightly coupled to a vendor-specific ACL API. Additionally, using an OpenFlow switch is a far more affordable than a fully- featured firewall that can forward at the speeds required for an exchange point.

5.1. Naïve Implementation

We created our prototype implementation[9] using the POX controller [12]. To receive BGP updates from the neighbors, we used ExaBGP [27] as a route server. Fig. 4 shows the implementation topology. In this example, ISP Z is the upstream ISP implementing the filtering at its exchange point with downstream ISPs (ISP A and ISP B).

The basic operation of our POX-based application generates rules using the following steps:

1) ExaBGP receives a new network advertisement from a neighbor.
2) The update goes into a first-in-first-out queue consumed by the POX app.
3) The POX app generates a rule for the OpenFlow switch that permits traffic sourced from the advertised network on the neighbor's switch-port.
4) Rules are removed whenever ExaBGP generates an expiration event as a result of the network no longer being advertised by that neighbor.
5) Traffic that doesn't match a rule is dropped.

[7]This is due to the reflecting DNS server belonging to a completely difference AS than the victim and the origin.
[8]Conversely, as the filtering network becomes more remote, the probability of filtering the attack packet also declines.
[9]The current source code is available at: http://homes.soic.indiana.edu/ktbenton/research/fprpf/

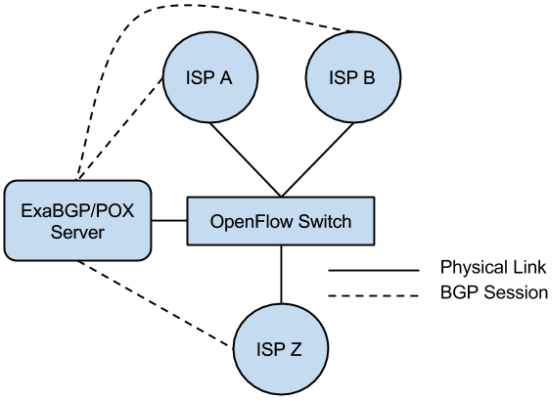


Fig. 4. Implementation topology.

With a few additional basic boilerplate rules to allow the BGP peering sessions, this is a simple implementation of feasible path filtering on an OpenFlow switch. However, it operates under the assumption that there will always be available space in the switch flow-table for more entries. While this is an acceptable assumption if peering with a few ISPs that only advertise a few hundred BGP prefixes each, this assumption fails in large exchanges or when one of the peers is providing transit for another ISP.

5.2. Summarization and Failing Gracefully

To handle the case where the required number of rules exceeds the flow-space available in the switch, we propose a few aggregation techniques followed by graceful failure modes when those aren't adequate.

1) Route Summarization: The first step is simple route summarization. It's possible that a neighbor will advertise multiple networks that, when combined, represent a larger CIDR block (e.g. 8.0.0.0/25 and 8.0.0.128/25). This occurs when the networks have differing AS path lengths so the neighbor can't summarize them on its own. The AS path length is irrelevant to our filtering so the networks can be safely combined into one CIDR.

2) Converting Allow Rules into a Mixture of Allow and Deny Rules: The second aggregation strategy is to convert many allows into a broader allow rule and higher priority explicit deny rules. For example, the group of updates [8.0.0.128/25, 8.0.0.0/26, 8.0.0.96/27, 8.0.0.80/28] can be turned into an allow rule for 8.0.0.0/24 with a deny rule for 8.0.0.64/28.

3) Leaky Aggregations: However, even with the above aggregation strategies, there needs to be process to handle the flow-table filling up. To handle this we propose a simple strategy to fail gracefully that does not unexpectedly drop packets. If the flow rules exceed the available space on the switch, we group the advertisements into "leaky aggregations" until the rules fit into the provided flow-table space.

We define a "leaky aggregation" as a summarization that includes an extra network that wasn't advertised by the neighbor. It's given this term because using this rule would "leak" any spoofed packets with a source from the extra network. This creates a "graceful failure" because it still prevents spoofing for the majority of the address space and it doesn't unexpectedly drop traffic.

The following Python code compresses a set of CIDRs into a "leaky aggregation":

The following is a more verbose description of the same process. First, place all of the advertised networks into a list sorted by descending netmask. Second, iterate through netmask values from 31 to 0, checking to see if adjusting a CIDR netmask to the new netmask value will eliminate the need for another CIDR. Then, each time an updated CIDR value is found that eliminates other values, update the CIDR set with the new values. Continue this process until the list falls within the required length (see Fig. 5).

By starting from netmask 31 and decreasing, the smaller leaky aggregations will be created before the large ones. By ending at 0, the program is guaranteed to return a list that passes the required traffic even if

it ends up being an allow rule for 0.0.0.0/0.

```
from netaddr import IPNetwork, IPSet
def leaky_compress(cidrs, max_entries):
  #Compresses cidrs into max_entries.
  #cidrs is a set of netaddr IPNetworks
  cidrs = sorted(
    cidrs, reverse=True,
    key=lambda x: x.prefixlen)
  if max_entries < 1:
    raise ValueError('Invalid max')
  netmask = 32
  while len(cidrs) > max_entries:
    netmask -= 1
    for cidr in cidrs:
      if cidr.prefixlen <= netmask:
        break
      copy = IPNetwork(cidr)
      copy.prefixlen = netmask
      new = IPSet([copy]) | IPSet(cidrs)
      new_cidrs = new.iter_cidrs()
      if len(new_cidrs) < len(cidrs):
        cidrs = set(new_cidrs)
        break
  return cidrs
```

Fig. 5. Python code for leaky aggregations.

With the two aggregation techniques above and the graceful failure mode, we believe that this is a reasonable prototype implementation of an FPRPF implementation designed not to impact normal traffic.

We have implicitly argued that an ISP should always advertise the source networks for the traffic it's sending to a peer, even if via a maximum length AS path. However, this is not necessary for a Tier 2 ISP to adopt the filtering as proposed. The design is incentive-aligned in that Tier 2 ISPs are more likely to be resource-constrained than Tier 1 ISPs, and thus have a greater interest in filtering. Tier 2 networks must pay for traffics, thus have an incentive to avoid being a major source in a DDoS. To the extent customers are harmed, Tier 2 networks have an incentive to protect these customers for being entangled in DDoS events. In addition, Tier 3 ISPs already advertise all of their downstream networks, thus no changes are required at most Tier 2/Tier 3 interfaces.

5.3. Impact on Dataplane Performance

The impact this filtering would have on the performance of the dataplane will be entirely dependent on the implementation of OpenFlow in the selected physical switches. Many implementations of the OpenFlow 1.x protocol map flows directly to a single table realized in ternary content addressable memory (TCAM) [28]. This allows a rule lookup to be performed in one clock cycle so traffic can be forwarded at the same speed as normal switching operations (a.k.a 'line rate'). Therefore, if a switch is chosen that implements OpenFlow in a TCAM (or similar performing mechanism), the impact to the dataplane would be as limited as a standard Ethernet switch used in Internet exchange points now.

6. Limitations and Future Work

A fundamental assumption in this work is that the filtering ISP can assume that all of its peering ISPs will advertise every network from which they source traffic. This is a reasonable assumption for Tier 2 ISPs with respect to peering Tier 3 ISPs. However, this is not a reasonable assertion facing Tier 1 or Tier 2 ISPs, both of which could have large complex networks with agreements to offload traffic from another ISP without

sending traffic or receiving BGP advertisements from said ISP. The traffic from these one-way transit agreements without advertisements will look like spoofed traffic and will be dropped.

6.1. Data Analysis on Traffic Flows

Investigation of the efficacy of this proposal filtering traffic from these larger ISPs is work in progress. In order to quantify how frequently our assumption is violated, we are working with PREDICT to obtain aggregated traffic flows to compare the source addresses with the networks found in the RIB outputs.

6.2. IPv6 Support

Another limitation is that our current implementation only supports filtering IPv4 networks. However, the high-level logic of filtering based on advertised prefixes will remain the same so it's just a matter of development time to add this support.

6.3. Aggregation Optimizations

OpenFlow offers various features that we haven't leveraged. Multiple flow-tables and non-contiguous bitmask matching could result in enforcement over a larger number of networks before entering the graceful failure mode.

6.4. Performing Alternative Actions and Adding Sources

We are currently working on incorporating external data sources beyond just the RIB to help augment decisions where the RIB does not provide enough information. Along similar lines, we will add extra actions beyond DROP so uncertain packets can just be logged, subjected to deep packet inspection, and/or downgraded to a lower quality-of-service.

7. Conclusions

In this work, we motivated the need for a new approach to the problem of IP spoofing. We then illustrated the limitations of the standard modes of reverse path filtering and presented an OpenFlow-based implementation of feasible-path RPF. Finally, we showed how our approach can scale by degrading gracefully without impacting the normal flow of traffic. We identified the ongoing work to quantify the impact this solution would have on ISPs if implemented.

As the next generation network is developed, many of the errors of the original control plane are being repeated. Physical security is assumed to cover network security; and all operators equally trusted. Individual security may be adequate for stand-alone systems but security in the control plane has unique requirements. One of these is incentive-align designed. Security mechanisms must be designed to be low cost and to provide primary first order benefits to adopting party. Security technologies cannot require ubiquitous adoption to be effective. Any solution that must be adopted by parties without the technical expertise, or by malicious parties, is doomed. Second order effects include system-wide results of partial adoption such as the effect of increased adoption on attacker cost, defender benefit, and defender cost. Using interdisciplinary analysis techniques we have illustrated the potential for incentive aligned design to secure not only endpoints but also to mitigate attacks that have proven intractable with current approaches.

Acknowledgment

Research was sponsored by the ARL under Cyber Security CRA W911NF-13-2-0045 (ARL); DHS Contract N66001- 12-C-0137, Cisco Research Proposal 591000 and Google Focused Research. The views and conclusions contained in this document are those of the authors and should not be interpreted as representing the official policies or views, either expressed or implied, of the DHS, ARL, DoD, Google, Cisco, IU, or the U. S. Government.

References

[1] Kantor, B. (1991). Rfc 1282: Bsd rlogin. *Internet Engineering Task Force*.

[2] Heberlein, L. T., & Bishop, M. (1996). Attack class: Address spoofing. *Proceedings of the 19th National Information Systems Security Conference* (pp. 371–377).

[3] Bellovin, S. (1996). Rfc 1948: Defending against sequence number attacks. *Status: INFORMATIONAL*.

[4] Kaminsky, D. (2008). Black ops 2008: It's the end of the cache as we know it. *Black Hat USA*.

[5] Rossow, C. (2014). Amplification hell: Revisiting network protocols for DDoS abuse. *Proceedings of Symposium on Network and Distributed System Security.*

[6] Gallagher, S. (2014). Biggest DDoS ever aimed at cloudflare's content delivery network. From http://arstechnica.com/security/2014/02/biggest-ddos-ever-aimed-at-cloudflares-content-delivery-network/

[7] Prince, M. (2013). The DDoS that knocked spamhaus offline. From http://blog.cloudflare.com/the-ddos-that- knocked-spamhaus-offline-and-ho

[8] Ferguson, P. (2000). Network ingress filtering: Defeating denial of service attacks which employ IP source address spoofing.

[9] Baker, F., & Savola, P. (2004). Ingress filtering for multihomed networks. BCP 84, RFC 3704, Tech. Rep.

[10] Cisco. (2005). Unicast reverse path forwarding commands. From http://www.cisco.com/c/en/us/td/docs/ios/122/security/command/reference/fsecurr/srfrpf.html

[11] Khouzani, M., Sen, S., & Shroff, N. B. (2013). Incentive analysis of bidirectional threat filtering in the internet.

[12] Mccauley, J. Pox: A python-based openflow controller. From http://www.noxrepo.org/pox/about- pox/

[13] Anderson, R. (1993). Why cryptosystems fail. *Proceedings of the 1st ACM Conference on Computer and Communications Security* (pp. 215–227). ACM New York, NY, USA.

[14] Camp, L. J., & Wolfram, C. (2000). Pricing security. *Proceedings of the CERT Information Survivability Workshop* (pp. 31–39).

[15] Varian, H. (2004). System reliability and free riding. In L. Camp & S. Lewis, (Eds.), *Economics of Information Security* (pp. 1–15). Springer US.

[16] Herley, C. (2009). So long, and no thanks for the externalities. *Proceedings of 2009 Work New Secur. Paradig Work* (p. 133). New York, USA: ACM Press.

[17] Dwork, C., & Naor, M. (1993). Pricing via processing or combatting junk mail. in E. Brickell, (Ed.), *Advances in Cryptology — CRYPTO 92* (pp. 139–147). Springer Berlin Heidelberg. From http://dx.doi.org/10.1007/3-540-48071-4 10

[18] Laurie, B., & Clayton, R. (2004). Proof-of-work? Proves not to work; version 0.2. *Proceedings of Workshop on Economics and Information Security*.

[19] Liu, D., &Camp, L. J. (2006). Proof of work can work. *WEIS*.

[20] Garg, V., & Camp L. J. (2013). Macroeconomic analysis of malware. *NDSS*.

[21] Adu-Oppong, F., Gardiner, C. K., Kapadia, A., & Tsang, P. P. (2008). Social circles: Tackling privacy in social networks. *Proceedings of Symposium on Usable Privacy and Security.*

[22] Moore, T., & Clayton R. (2008). Evaluating the wisdom of crowds in assessing phishing websites. *Financial Cryptography and Data Security*, 16–30.

[23] Vasek, M., & Moore, T. (2012). Do malware reports expedite cleanup? An experimental study. *Proceedings of the 5th USENIX Conference on Cyber Security Experimentation and Test* (p. 6). USENIX Association.

[24] Feamster, N., Rexford, J., & Shenker, S. SDX: A software-defined internet exchange.

[25] Stringer, J. P., Fu, Q., Lorier, C., Nelson, R., & Rothenberg, C. E. (2013). Cardigan: Deploying a distributed

routing fabric. *Proceedings of the Second ACM SIGCOMM Workshop on Hot Topics in Software Defined Networking* (pp. 169-170). ACM.

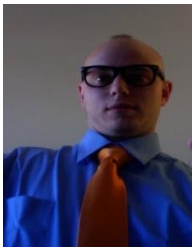
[26] McKeown, N., Anderson, T., Balakrishnan, H., Parulkar, G., Peterson, L., Rexford, J., Shenker, S., & Turner, J. (2008). Openflow: Enabling innovation in campus networks. *ACM SIGCOMM Computer Communication Review*, *38(2)*, 69–74.

[27] Exabgp: The bgp swiss army knife of networking. From https://github.com/Exa-Networks/exabgp

[28] Corporation, B. (2014). Engineered elephant flows in large scale clos networks. From https://www.broadcom.com/collateral/wp/OF-DPA-WP102- R.pdf

Kevin Benton is a PhD student in the School of Informatics and Computing at Indiana University. He received his master degree in security informatics from the University of Nevada, Las Vegas and his B.S. degree in network technology from Montana Tech of the University of Montana. His research interests include network security, software-defined networking, and applied cryptography.

L. Jean Camp is a professor at the School of Informatics and Computing at Indiana University. She joined Indiana from Harvard's Kennedy School after a year as a senior member of the technical staff at Sandia National Laboratories. She has a doctorate from Carnegie Mellon, and MSEE from UNC-Charlotte. She has authored more than one hundred fifty publications, including more than one hundred peer-reviewed publications and multiple books. She has peer-reviewed publications on security and privacy at every layer of the OSI model.



Timothy Kelley is a visiting research at the Developmental Cognitive Neuroscience Lab at Indiana University, Bloomington. He received his PhD degree in cognitive science and informatics from the Department of Psychology and Brain Sciences and the School of Informatics at Indiana University, Bloomington for analyzing systemic effects of small-scale behavior in regards to the economics of information security. He is currently researching continuous dynamics of real-time decision-making and stochastic modeling of cognition, behavior, and networks.

Martin Swany is an associate professor of computer science at the School of Informatics and Computing at Indiana University. He received his PhD degree in computer science from the University of California, Santa Barbara. His research areas include high performance computing, distributed computing, computer networks, and compilers. Additionally, he is the associate director of the Center for Research in Extreme Scale Technologies.