Protecting BGP Routes to Top Level DNS Servers

Lan Wang[1], Xiaoliang Zhao[2], Dan Pei[1], Randy Bush[3], Daniel Massey[2],
Allison Mankin[4], S. Felix Wu[5], Lixia Zhang[1] *

Abstract

The Domain Name System (DNS) is an essential part of the Internet infrastructure and provides fundamental services, such as translating host names into IP addresses for Internet communication. The DNS is vulnerable to a number of potential faults and attacks. In particular, false routing announcements can deny access to the DNS service or redirect DNS queries to a malicious impostor. Due to the hierarchical DNS design, a single fault or attack against the routes to any of the top level DNS servers can disrupt Internet services to millions of users. In this paper we propose a path-filtering approach to protect the routes to the critical top level DNS servers. Our approach exploits the high degree of redundancy in top level DNS servers and also exploits the observation that popular destinations, including top level DNS servers, are well connected via stable routes. Our path-filter restricts the potential top level DNS server route changes to be within a set of established paths. Heuristics derived from routing operations are used to adjust the potential routes over time. We tested our path-filtering design against BGP routing logs and the results show that the design can effectively ensure correct routes to top level DNS servers without impacting DNS service availability.

Keywords: fault-tolerance, DNS infrastructure protection, route hijacking, BGP path filtering

1 Introduction

The Domain Name System (DNS) [9] is an essential part of the Internet infrastructure. It provides the service of translating host names, such as www.cs.ucla.edu, into IP addresses that are used for data delivery. If an application fails to receive a reply for its DNS query, it is denied service. Worse still, if an application receives a reply that contains a wrong IP address, it will send data either to a black hole or to a machine selected by an attacker. Due to its hierarchical design, failure to reach all the 13 DNS root servers would cripple the entire DNS service and make all destinations unreachable by most applications. This potential vulnerability of the root servers is well known and has even been described in popular press articles [8]. In addition to the root servers, there are also 13 DNS servers for the generic top level domains (gTLDs) including com, net and org. The loss of reachability to these gTLD servers would also deny access to millions of destinations in com, net, and org name domains. In today's Internet, announcing a false route to DNS servers can easily lead to such faults or attacks.

To assure reliable service, the 13 DNS root servers are located in diverse and well-connected parts of the network. A recent ICANN report suggests that the DNS system can continue to operate correctly even when 5 of 13 root servers are unavailable due to faults or attacks [4]. However, the report overlooked the impact that network routing faults might have on the reachability to the root servers. Although the top level DNS servers themselves are closely monitored to guard against compromises, a sophisticated attacker can bypass current security measures by inserting a false network route that redirects DNS queries from an intended DNS server to the attacker. Since secure DNS is not deployed at this time, the lack of authentication in today's DNS service allows the attacker to reply with any desired DNS response. Thus even a single false route announcement for a top level DNS server can have catastrophic consequences impacting millions of Internet hosts.

Measurements have shown that faults in BGP, the current global routing protocol, do occur from time to time [7]. Furthermore, our analysis shows that invalid BGP routes to root and gTLD DNS servers have indeed occurred in the Internet. For example, BGP routing logs from RIPE[13] show that on April 26, 2001 an AS incorrectly advertised a route to the "C" gTLD server. A number of large (Tier-1) ISPs adopted this false route for several hours; during that time period, all DNS queries to the "C" gTLD server sent by clients in those ISPs were supposedly delivered to the faulty AS. In this case, the false route was due to a misconfiguration. However, note that an intentional attack could

* [1]Lan Wang, Dan Pei and Lixia Zhang are with UCLA. E-mail: {lanw, peidan, lixia}@cs.ucla.edu [2]Xiaoliang Zhao and Daniel Massey are with USC/ISI. E-mail: {xzhao, masseyd}@isi.edu [3]Randy Bush is with IIJ. E-mail:randy@psg.com [4]Allison Mankin is with Bell Labs. [5]S Felix Wu is with UC Davis. E-mail: wu@cs.ucdavis.edu

have directed these queries to a false server and that server could have replied with any desired data.

In this paper we propose a path-filtering mechanism to protect the universally critical BGP routes that lead to the root/gTLD DNS servers. Our design makes use of two basic facts. First, increasing evidences show that popular destinations have relatively stable routes [12]. Our analysis also shows that the root/gTLD DNS servers can be reached through a set of stable routes (see Section 3). Second, the root/gTLD DNS servers have a high level of redundancy, therefore the DNS service will not be impacted by a temporary loss of reachability to one of the servers. The first fact enables our path-filtering mechanism to use a heuristic approach to identify potentially valid routes, while the second fact provides tolerance for occasional errors when path-filtering rejects a valid route. We have tested our path-filtering design against 12 months of BGP routing logs. Our results show that, after applying our path filter, root/gTLD server reachability remains high while invalid routes are successfully blocked by the filter.

Due to its critical importance, there has been an increasing effort over the last few years to add security to the DNS service [1]. However, the authentication of all DNS query replies does not address the potential problem of DNS queries being hijacked through false route announcements which could lead to denial of service attacks. The work presented in this paper explores a new venue in building a resilient Internet infrastructure. Our approach utilizes certain properties from the network infrastructure to guard against faults and attacks. Compared to cryptography-based mechanisms, our path-filtering mechanism has an advantage of being readily and incrementally deployable. Overall, our approach is different from and complementary to the conventional security solutions that rely on cryptography-based security mechanisms.

In the rest of the paper, Section 2 provides background information on DNS and BGP. Section 3 examines the stability of root/gTLD server routes. Section 4 describes a simple path filter example for root/gTLD server routes protection. In section 5, we present our adaptive path filter design. Section 6 evaluates our path-filtering design by applying the scheme against BGP updates from several diverse ISPs. Section 7 presents the related work and Section 8 summarizes the contribution of our work.

2 Background

2.1 DNS Overview and Terminology

The Domain Name System is a distributed database that maps host names to IP addresses and provides other fundamental information. The DNS name space is organized in a tree structure, as shown in Figure 1. A DNS resolver

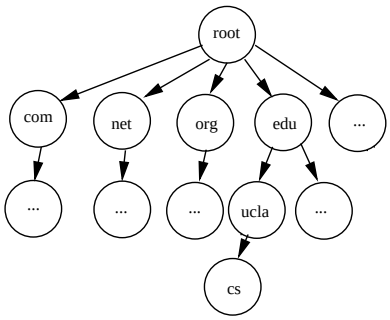


Figure 1. The DNS Name Space Tree Structure

requests data by first querying one of the root servers. Using the referral information from the root server, the resolver proceeds down the tree until the desired data is obtained. For example, a resolver seeking the IP address of *www.ucla.edu* starts by querying any root server and the root server provides a referral to the DNS servers for the *edu* domain. A query to any of the *edu* DNS servers returns a referral to the *ucla.edu* servers and finally a query to one of the *ucla.edu* servers returns the IP address for *www.ucla.edu*. In practice the process is slightly more complex; a resolver typically queries a local DNS server and the local server performs the query starting from the root.

To both protect against faults and help distribute the load, each zone in the DNS tree should operate multiple DNS servers in diverse parts of the network. For the root zone, there are 13 root servers and each has an identical copy of the DNS root zone[1]. There are also 13 gTLD servers that serve three top level domains com, net and org. The DNS system can withstand the loss of some top level servers since if one server fails to reply, the resolver simply tries the other replicate servers. However, hijacking DNS queries to even a single DNS root/gTLD server can have catastrophic consequences. The DNS service, as currently deployed, has no authentication mechanism other than a query ID number chosen by the resolver sending the query. Any response with a matching query ID is accepted. Our goal in this study is to add a simple protection mechanism to guard the DNS service against faults and attacks through false routing announcements.

2.2 BGP Overview and Terminology

The Internet is divided into thousands of Autonomous Systems (ASes), loosely defined as networks and routers under the same administrative control, and BGP[11] is the de facto inter-AS routing protocol. A BGP route lists a

[1] Size limitations in the DNS protocol[9] restrict the maximum number of root servers to 13.

particular prefix (destination) and the path of ASes used to reach that prefix.

In this paper we are only concerned with the validity of the BGP routes to top level DNS servers. A DNS server is considered reachable as long as the BGP route remains in place even though the server might be down.

3 Top-level DNS Server Routes

In this section, we analyze BGP log data to verify one important design criteria: the stability of the BGP routes to the root/gTLD DNS servers.

3.1 Data Source

Our data set contains BGP updates collected by the RRC00 monitoring point at RIPE NCC [13] from February 24, 2001 to February 24, 2002. The monitoring point peers with ISPs which have a wide range of operation characteristics (see Table 1). The nine peer routers are located in US, Japan and three European countries. Some of them belong to large global ISPs and others belong to regional ISPs. Before conducting the analysis, we carefully removed the known measurement artifacts from the collected data using the technique described in [15].

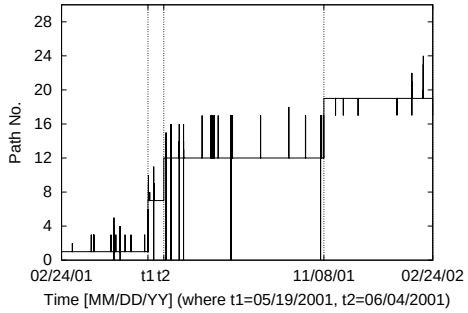
Location	ASes that RRC00's peers belong to
US	AS7018 (AT&T), AS2914 (Verio)
Netherlands	AS3333 (RIPE NCC), AS1103 (SURFnet), AS3257 (Tiscali)
Switzerland	AS513 (CERN), AS9177 (Nextra)
Britain	AS3549 (Global Crossing)
Japan	AS4777 (NSPIXP2)

Table 1. RRC00's Peering ASes

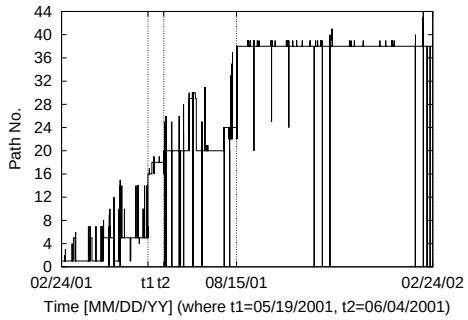
For each DNS root/gTLD server, we identified the longest address prefix that covers the server's IP address and extracted the BGP updates for these prefixes from the archive. It should be noted that the "A" and "J" root servers share the same address prefix 198.41.0.0/24 in the BGP routing table.

3.2 Routing Stability

Figure 2 shows ISP1 and ISP2's AS paths to the A root server. The x-axis is time and y-axis is the number assigned to each specific AS path. For example, (x_1, y_1) indicates that path y_1 is being used at time x_1, and a shift from (x_2, y_1) to (x_2, y_2) means the path to reach the A root server changed from y_1 to y_2 at time x_2. In addition, a path labeled 0 means there is no route to the root server. The figures show that, during this one year period, each of the two



(a) ISP1



(b) ISP2

Figure 2. AS Paths to the A Root Server

ISPs used a small number of primary AS paths to reach the A root server. In particular, the figure shows that one primary path was used most of the time; when the primary path was unavailable, a few alternative paths were used for very short time periods. Moreover, the primary AS path usually lasted weeks or even months before the ISP switched to a new primary path. The other 7 ISPs peering with the monitoring point also show similar stability in their routes to the root/gTLD servers; their figures were omitted for brevity.

To better understand how the characteristics of the routes to reach the DNS servers shape our design, we present ISP1's AS path changes to the A root server in more detail. The 3 dotted lines in Figure 2(a) divide the graph into four regions; a different primary path is used in each region. The first region begins on February 24, 2001 and lasts until May 19, 2001. During this period, ISP1 used path 1, $(7018, 4200, 2645)$, to reach the A root server except for 1.7 hours when ISP1 used a few other paths. The figure also shows two instances when ISP1 had no path to reach the A root server: a 28-second period on April 16, 2001 and a 157-second period on April 22, 2001.

On May 19, 2001, ISP1's primary AS path to

the A root server changed from $(7018, 4200, 6245)$ to $(7018, 10913, 10913, 10913, 10913, 11840)$; the old path never reappeared. The origin AS for the A root server's address prefix was changed from AS6245 to AS11840, both AS6245 and AS11840 are owned by the same organization which runs the A root server. This origin AS change reflected an operational change. The AS path $(7018, 10913, 10913, 10913, 10913, 11840)$ became the new primary path and it exhibited the same stable behavior as the previous one, being used almost all the time till June 4, 2001. During the few occasions when some alternative route was used, none of them lasted more than 15 minutes.

On June 4, 2001, the origin AS for the A root server changed again and $(7018, 10913, 10913, 10913, 19836)$ became the new primary path. Again this change reflected an operational change by the site running the A root server. A third change occurred on November 8, 2001 when a transit AS, AS 10913, changed its routing policy. AS10913 stopped listing its own AS number multiple times and the new primary path became $(7018, 10913, 19836)$.

In summary, the primary AS path to the A root server changed 3 times over the 12-month period, twice because of a change in the origin AS and once because a transit AS changed its routing policy. Assuming some means existed to adjust the path filter following these changes, any router peering with ISP1 could apply a simple filter containing the primary AS path to protect its route to the A root server. Its reachability to the A root server might be impacted slightly, but it would have gained the protection against any invalid routes to the A root server.

4 A Simple One-Path Filter

To illustrate the protection power of path-filtering, we first consider using a *single* route as the filter. In other words, we assume that an ISP selects one allowable AS path for each of the 13 DNS root/gTLD servers and rejects any other paths it receives from BGP route advertisements. Because a high level of redundancy is built into the root/gTLD server implementations, the simple filter's lack of adaptability to transient route changes does not necessarily impact DNS service availability.

As we have seen in the previous section, network topology and routing policies do change over time and can lead to long term changes to the primary paths for the top level DNS servers. Therefore, although the one-path filter does not adapt to transient route changes, the filter must adapt to long-term changes. Note that the servers' IP addresses may also change, but the changes are very rare as they affect millions of users. Changes are also widely announced ahead of time, which gives ISPs enough time to adjust their DNS server and filter setting.

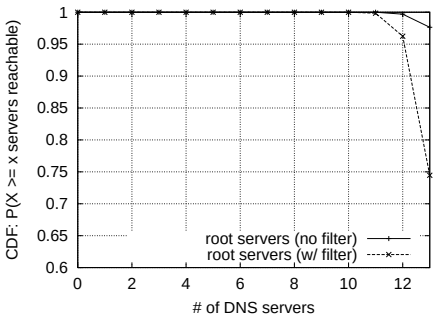


Figure 3. Overall Root Server Reachability

In the rest of this section, we estimate an upper bound of the one-path filter's performance assuming that ISPs are always able to adjust their filters to accommodate long term routing changes at the right time. More specifically, we divided the 12-month data into week-long segments and, for each week, we set the filter to be the primary AS path of that week.

4.1 Root Server Reachability

We applied the one-path filter to each ISP's BGP updates. Because these BGP updates contain the routes that an ISP chose to advertise its peers rather than the routes that the ISP received from its peers, we are in effect simulating the filter's effect on a BGP router which has the ISP as its single peer (e.g. one of the ISP's single-homed customers). Due to space constraint, we present the results for ISP1 (the other ISPs have similar results).

The graph in Figure 3 shows the effect of the filter on the simulated router's overall reachability to the 13 root servers. Before applying the filter, the router has paths to reach all the 13 servers about 98% of the time, and can reach at least 12 servers nearly 100% of the time. After applying the filter, it has paths to all the 13 servers about 77% of the time, and can reach at least 12 servers about 96% of the time. Most importantly, it can always reach at least one server with the filter. In summary, the results indicate that reachability can be extremely high if the filter can be adjusted to accept long-term path changes.

4.1.1 Slow Convergence

After a topology or routing policy change, BGP explores the (potentially very large) set of all possible routes before converging on a new stable route or declares the prefix is unreachable. Previous measurement [6, 10] showed that this convergence delay may last 3 minutes on average, and some non-trivial percentage of cases lasted up to 15 minutes. The transient paths during the slow convergence period do not necessarily provide reachability to the destination.

We examined the BGP updates to determine how many back-up routes might have been the result of BGP slow convergence. Between Feb. 24, 2001 and May 19, 2001, there were 15 times when ISP1's primary AS path to the A root server was replaced by a back-up route. The majority of the back-up routes remained in place for only seconds. In only 6 out of the 15 instances the back-up route remained in place for longer than 3 minutes, and only 1 back-up route lasted longer than 15 minutes. With our limited data set, we could not confirm which routes were valid back-up routes and which were simply an artifact of BGP slow convergence. However we speculate that those short-lived back-up routes are most likely invalid ones. By rejecting these false back-up routes, the one-path filter would not decrease the actual reachability to the root server and could actually help stop the propagation of transient routes.

4.2 Limitations of the One-Path Filter

Overall, the ISPs examined in our study seem to use a single primary path to reach each top level DNS server. Thus if a router peering with any of the ISPs had applied the simple path filter as a protection measure, it would have maintained a high level of reachability to the DNS root servers. However, one of the ISPs performed much worse than the others. If a router peers with this ISP, then the reachability to *all* the 13 root servers is nearly 100% of the time without the filter, but drops to only 35% of the time after applying the filter. Although the router could still reach at least 1 root server 99.97% of the time, the decrease in reachable servers raises a concern. By analyzing the BGP log data we observed that this ISP uses one primary path and a small number of consistent back-up paths to reach some of the root servers. This fact suggests that, in addition to trusting a primary path, we must enhance our one-path filter with several allowable back-up paths to provide both strong route protection and high service availability.

As Section 3.2 shows, updates to the primary path are also needed from time to time. An ideal filter would automatically detect the primary path changes and update the filter accordingly. In the next section we extend the one-path filter to a design that can support multiple valid paths as well as automated filter updates.

5 Adaptive Path Filter Design

The simple one-path filter is effective in filtering out invalid paths, but it allows only a single route to each root/gTLD server and requires occasional manual updates when the primary AS path changes. In this section, we present a more advanced filter that maintains a set of potentially valid paths for each root/gTLD server and automatically includes new valid routes that result from topology or routing policy changes. We use both route history and strong validation mechanisms to identify new valid paths.

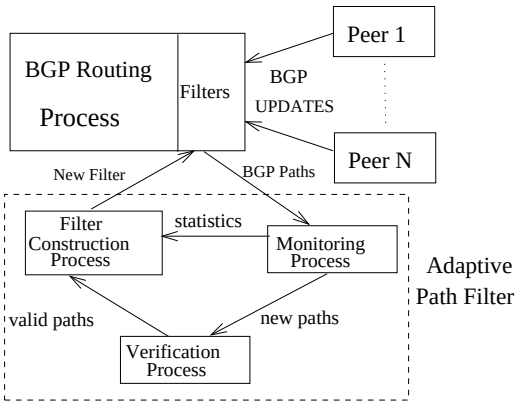


Figure 4. Adaptive Path Filter Design

5.1 Design Overview

Our path filter has three components (see Figure 4).

- A *monitoring process* is used to identify potential new paths and keep track of route history.
- A *verification process* is used to validate the new path and periodically re-verify existing paths.
- A *filter construction process* dynamically adjusts the filter for a root/gTLD server based on the BGP path statistics collected by the monitoring process and the feedback from the verification process.

To set up a path filter for a DNS server, we first identify the address prefix that covers the IP address of the server, and then choose a set of initial paths that can be used to reach the server. This initial choice is based on past experience and known routing policies. The filter algorithm then adjusts path filters at time intervals of length T in order to accommodate dynamic changes in topology and routing policy.

During each time period, the router monitors the percentage of time a path is announced by a peer, regardless of whether the path is currently in a filter. At the end of a time period, we consider only paths that have been announced more often than a configured threshold. The threshold is used to screen out misconfigured and transient paths; those paths whose usage exceeds the threshold are called *base paths*. The verification process is then invoked to check whether the base paths are indeed valid. All previously adopted paths are also verified again with a probability P_v. Only paths that have appeared as base paths and that have passed the verification test will be placed in the filter set for the next time period.

If a new legitimate route appears at the beginning of a time period, the route could be delayed for T amount of time before being added to filter. Therefore, as an enhancement to the above algorithm, we also adopt *valid* paths into the path filters if they have been advertised for a period of T_r in the current time period. T_r is typically much smaller than T and, an appropriate value of T_r should ensure quick adaptation to topology changes while minimizing the impact of transient paths. These paths also need to be validated by the verification process before they can be added to the filter set. If these paths do not meet the criteria for base paths, they will be eliminated from the path filters at the end of the time period T. Sections 5.2–5.4 describe the algorithm in more detail.

5.2 Monitoring Process

The monitoring process collects path statistics. In the $k'th$ time period, it keeps track of $T_k(p)$, the amount of time each path p is used by a peer to reach the DNS server D. At the end of the time period, $T_k(p)$ will be used by the filter construction process to construct the filter set for the next time period.

In addition, if p is not in the current filter set and if $T_k(p)$ exceeds a threshold T_r, then the verification process is invoked to determine whether p should be added to the current filter set. If the verification process determines that the path is valid, p is immediately placed in the filter set.

5.3 Filter Construction Process

At the end of each time period, a new filter set is constructed based on the old filter according to the following steps. Figure 5 shows the algorithm in pseudo code.

First, each path's usage is calculated as $U_k(p) = T_k(p)/T$ where $T_k(p)$ is the value provided by the monitoring process and T is the length of the time period. Paths with a $U_k(p)$ above a minimum threshold will be considered as base paths and are candidates for inclusion in the next filter set. However, if a path is oscillating between extreme values of $U_k(p)$, then creating the new filter set based solely on this measure can cause a valid path to be moved in and out of the filter set at the wrong times. To account for this problem, the filter construction also uses $U(p)$, an exponentially weighted moving average (EWMA) of $U_k(p)$, to select base paths. $U(p)$ is calculated using the following formula

$$U(p) = (1 - \alpha) * U(p) + \alpha * U_k(p), \qquad (1)$$

where $\alpha \in (0, 1)$.

A path is considered as a base path only if $U(p)$ or $U_k(p)$ is greater than the minimum threshold U_{min}. For any new base path, the verification process will be invoked. For base

```
while in_current_time_period()
      %Receive new paths from verification process
      p = recv_new_path();
      F = F ⋃ p;
      p.checked = 1;
end
%Adjust F at the end of a time period
F_old = F;
F = ∅;
foreach p ∈ F_old
  if U(p) ≥ U_min ∨ U_k(p) ≥ U_min
    %Consider only base paths
    then
        if p.checked = 1
          then
              %p is already validated
              F = F ⋃ p;
           else
              %Validate p with a probability of P_v
              if rand() ≥ P_v ∨ verify(p) = 1
                then F = F ⋃ p;
              fi
        fi
  fi
  p.checked = 0;
end
```

Figure 5. Algorithm to Adjust a Filter

paths that have been verified in the previous intervals, the verification process is invoked with a probability of P_v. Any base path that fails the verification process is rejected and the remaining base paths form the new filter set.

5.4 Verification Process

Verification can be performed by either humans or automated programs. The separation of the monitoring and verification functionality in our design allows each site to experiment with different approaches to verification and improve them based on experience. The simplest method is to inspect the BGP path for anomalies such as reserved AS numbers and perhaps use the IRR (Internet Routing Registry) to verify if the origin AS is authorized to originate the prefix. However, this method is not reliable because the BGP path may not contain obvious anomalies and the records in IRR are often inaccurate.

We propose an automated verification process that utilizes the redundancy of the DNS system. The basic idea is to send a DNS request to the DNS server on the suspected path and validate the reply against answers from other DNS servers for the same request. Note that, to ensure that the request will travel on the suspected path, we may execute the verification process only when the peer is still using the

path. If the suspected path is injected accidentally, no reply may be received. Otherwise, a reply may come from an impostor (i.e. a false DNS server set up by the attacker who injected the suspected path) and contradict that of the other DNS servers.

The impostor may try to defeat our verification process by giving correct answers, especially at the beginning of their attack. However continuous re-verification of paths should allow us to catch the impostor eventually (see Section 5.3).

We believe that path verification is itself an important research area and is part of our future work. The focus of this work, however, is to explore the feasibility of path filtering, design the general framework and build a prototype.

5.5 Parameter Setting

The time period T may be on the order of days, weeks or even longer depending on the desired level of security and the projected overhead. In general, a longer time period means less frequent verification. So, while a long T can keep the overhead low, it may result in weaker protection. T is set to one week in our experiments.

A larger U_{min} may result in smaller path filters since fewer paths will be considered as base paths. This may lead to lower reachability to the servers, but it also provides stronger protection against temporary or misconfigured paths. One can therefore select an appropriate value based on the required level of reachability and the desired level of security. We choose a U_{min} of 10% in our study, i.e., if a path is used cumulatively for more than 10% of the time in a time interval, it will be considered a base path.

Using an EWMA of $U(p)$ allows a path to stay in the filter for a period of time even if it is not used by the peer. Suppose path p is now obsolete and its current $U(p)$ is 1, then we will keep this path for a period of $\lceil \log(U_{min}) / \log(1 - \alpha) \rceil \times T$ before eliminating it from the filter. Although we would still verify this path with a certain probability during this interval, it is possible that, before we detect the path is no longer valid, a malicious attacker injects this path into the network. To minimize the risk of accepting such a spoofed route, we could use a larger α, a larger U_{min} or a smaller T. However, the trade-off is that we may prematurely remove paths that will soon return to use.

6 Evaluation

In this section, we again use the BGP updates collected at RRC00 from Feb. 24, 2001 to Feb. 24, 2002 to evaluate the adaptive path filter. We simulate a router that peers with one of the ISPs in Table 1 and compare its routes to the root/gTLD DNS servers before and after filtering. The filter parameters we use are shown in Table 2.

T	T_r	U_{min}	α	P_v
1 week	1 hour	10%	0.25	0.1

Table 2. Parameter Setting

For this study, the monitoring process identifies and reports to the verification process any new paths that exist long enough to warrant further checking, but since we are using archived log data we cannot directly apply a verification process to the new paths (the proposed verification mechanism works with paths that are still in use by the peer). Therefore, we assume the paths were accepted as valid. It remains part of the future work to evaluate the verification mechanism on peering sessions with operational BGP routers. However, note that the focus of this paper is not the verification process, but rather the feasibility of the path filtering approach.

Our results show that, even without a verification mechanism, the filter is able to reject many transient paths while adapting to long-term path changes. For example, the simulated router filtered out most of the short-lived back-up paths to the A root server that were announced by ISP1 and automatically adapted to the three actual routing changes described in Section 3.2. As another example, over time ISP2 announced a total of 44 different paths to the A root server, seven of the 44 are stable paths. When our router peered with ISP2, it screened out 25 of the 44 paths and all the 7 stable paths were adopted as valid routes in the filter. In the remainder of this section, we first examine the nature of those paths that were filtered out and then present the filter's impact on root/gTLD server reachability.

6.1 Filtering Invalid Routes

In this section, we show that the paths rejected by our filter are mainly invalid paths such as those involved in Multiple Origin AS (MOAS) conflicts due to operation errors [16] or are routes that appear during slow BGP routing convergence.

6.1.1 Invalid Routes Due to Misconfiguration

On April 6,2001, an AS mistakenly originated a false path to DNS "C" gTLD server. Out of the 9 ASes we observed, 4 of them selected this bogus path as the best path to reach "C" gTLD server. However when the filter is applied to the BGP updates, all but one of the false announcements by the faulty AS were blocked out; that one skipped path lasted for more than 3 hours and was forwarded to the verification process.

Time	BGP Path
12:35:30	3549 19836 19836 19836 19836
16:06:32	3549 10913 10913 10913 10913 10913 10913 10913 19836
16:06:59	3549 1239 10913 19836
16:07:30	3549 701 10913 10913 19836
16:08:30	Path Withdrawn
16:15:55	3549 19836 19836 19836 19836

Table 3. A Slow Convergence Example

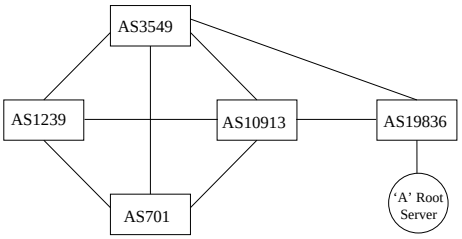
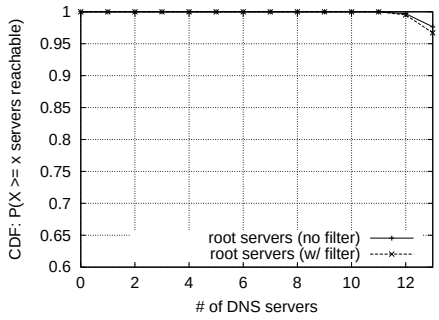


Figure 6. Slow Convergence Topology

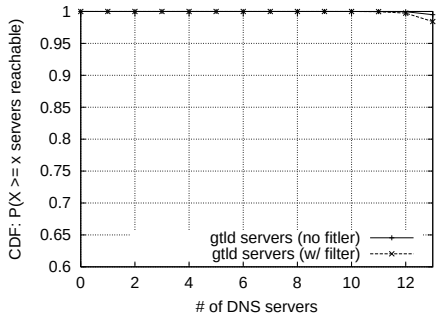
6.1.2 BGP Slow Convergence

Table 3 shows the BGP update sequence in a typical BGP slow convergence scenario (see Figure 6 for the topology of the ASes involved the slow convergence). On June 19, 2001, AS3549 was using the path $(3549, 19836, 19836, 19836, 19836)$ to reach the DNS A root server. From this AS path, we can see that AS19836 originated this path and AS3549 directly peers with AS19836. At 16:06:32, AS3549 announced another path to the A root server, implying it could no longer reach the server through the direct peering with AS19836. This type of path change might be due to a break down in the link between AS3549 and AS19836, but BGP logs show that several ISPs that did not depend on the AS3549-AS19836 connectivity also lost their routes to the A root server at the same time. Thus it is most likely that the problem was inside AS19836. Nevertheless AS3549 continued in vain to explore all possible paths through its other neighbor ASes and generated additional closely spaced new path announcements. After trying all its back-up routes, AS3549 finally sent a path withdrawal message.

BGP log data showed that this incidence of slow convergence produced 24 unnecessary BGP updates from the nine peers and our filter was able to filter out all of them. In other words, if deployed in the current Internet, the filter would have effectively blocked these false updates from further propagation, thus reducing the processing load at BGP routers and possibly improving the routing convergence time.



(a) root servers



(b) gTLD servers

Figure 7. Reachability through ISP1

6.2 Impact on Server Reachability

In this section, we compare the difference in server reachability with and without using the filter. Ideally, the use of filter should cause negligible decrease in reachability. We first show the results for ISP1, then we summarize results for the other ISPs.

Figure 7(a) shows the reachability to the DNS root servers through ISP1. The x-axis is the number of servers and y-axis is the percentage of time when x or more root/gTLD servers are reachable. The solid and the dashed curves correspond to the reachability before and after we applied the filter, respectively. The two curves overlap until x is 10. For x equal to 11, there is a very small difference: 99.997% without filtering and 99.982% with filtering. The difference becomes 0.948% for x equal to 13 root servers. It is evident that the filter has little impact on the router's reachability to the root servers. Since the graph for the gTLD servers (Figure 7(b)) is quite similar to that for the root servers, we do not discuss it in detail here.

The results for all the nine ISPs are summarized in Table 4. We only present the reachability to the root servers

No. Servers	N=1		N=6		N=13	
	Unfiltered	Filtered	Unfiltered	Filtered	Unfiltered	Filtered
ISP1	100%	100%	100%	100%	97.649%	96.701%
ISP2	99.998%	99.974%	99.972%	99.939%	99.398%	97.039%
ISP3	100%	100%	100%	100%	95.149%	93.974%
ISP4	100%	100%	99.940%	99.855%	98.017%	95.981%
ISP5	100%	99.978%	99.951%	99.947%	91.341%	90.228%
ISP6	100%	100%	99.397%	99.397%	98.808%	97.248%
ISP7	100%	100%	99.998%	99.966%	99.535%	97.395%
ISP8	100%	99.996%	99.975%	99.971%	25.728%	25.419%
ISP9	100%	100%	99.994%	99.992%	99.475%	97.780%

Table 4. Percentage of Time when N or More DNS Root Servers are Reachable

here (the results for the gTLD servers are similar). We make the following observations:

1. One could reach at least one root server through six of the nine ISPs 100% of the time after we applied the filter. The same statistic for the other three ISPs is 99.974% for ISP2, 99.978% for ISP5 and 99.996% for ISP8.

2. One could reach at least 6 root servers through either ISP1 or ISP3 100% of the time even after we applied the filter, while the same statistic for the other ISPs ranges from 99.855% to 99.992%. This is because ISP1 and ISP3 (both are US-based global ISPs) had considerably better reachability to the root servers before filtering was applied (see Figure 7).

3. The percentage of time to reach all the 13 root servers through ISP8 is very low (25.728%). This is mainly due to ISP8's poor reachability to the DNS "E" root server: it had no route to this server 71.5% of the time during our study period. The percentage of time when all the 13 servers were reachable through the other eight ISPs ranges from 90.228% to 97.780%.

It is essential that a network be able to reach at least one DNS root/gTLD server at all times in order for DNS to operate correctly. As we have seen, six of the nine ISPs we studied allow our router to satisfy this requirement after adding the filter protection. If a network has less than ideal reachability to the DNS servers even without the filter, it needs to improve its reachability first.

It is also important to keep in mind that we measured the reachability through a *single* ISP, but in reality an ISP peers with multiple ISPs and multi-homed client ASes are becoming a common case. Thus we expect the actual reachability to top level DNS servers to be much higher than the values we report here, both before and after the deployment of our path-filtering protection mechanism.

7 Related Work

A detailed analysis of potential threats to DNS service can be found in [2]. DNSSEC [1] is designed to detect spoofed DNS replies. It uses public key cryptography to authenticate data origin and ensure data integrity. The verification process depends on a chain of trust that starts with the assigned key for the root zone and proceeds through the hierarchical resolution of a domain name.

DNSSEC adds considerable complexity, space and processing overhead into the network [2]. Its deployment is also a challenging undertaking which will take a long time. On the other hand, our path-filtering mechanism can be deployed by ISPs today. Furthermore, even full deployment of DNSSEC does not eliminate the need for protecting the routes leading to DNS servers. However, it can detect a false response if a DNS query is hijacked to an attacker's machine, therefore it can serve as one of the validation steps mentioned in Section 5.4. In summary, the path-filtering mechanism and DNSSEC are complementary approaches to protecting DNS service.

Several anti-route-spoofing approaches have been proposed previously. Notably [5] proposed the use of a public key infrastructure to verify each route advertisement. However, this approach calls for significant changes to the current Internet routing infrastructure. The "predecessor" and path finding approach proposed by [14, 3] can be used to authenticate the AS path, but it cannot prevent an AS from falsely originating a route to a prefix it cannot reach. [17] proposed a protocol enhancement that enables BGP to distinguish false route announcements from valid ones. The design utilizes the fact that the Internet is a richly interconnected system, making it difficult for any fault or attack to completely block correct routing information from propagating through. However, the approach as described in [17] can detect only certain types of false route announcements. As BGP security evolves forward, we believe the path-filtering mechanism will continue to play a comple-

mentary role in protecting critical DNS servers as an added line of defense against faults and attacks.

8 Conclusion

The Internet is a large-scale, distributedly managed, and still rapidly growing infrastructure system. Faults and attacks are inevitable events in such large scale systems. In this paper we present a path-filtering mechanism to guard the DNS root and gTLD service against faults and attacks in network routing. Our design exploits the high degree of redundancy in the top level DNS system and the stability in network connectivity to the server locations. The proposed mechanism filters out potentially invalid routes by restricting the routes to the top-level DNS servers to change within a set of established routes. The set of established routes can adapt to long-term topology changes by adding only verified persistent routes. We have evaluated the feasibility and effectiveness of path-filtering mechanism using 12 months of BGP route logs. The results show that our design effectively detects the insertions of invalid routes with little impact on DNS service reachability. Furthermore, the path-filtering mechanism is simple and readily deployable by ISPs.

The work presented in this paper is one of the first steps toward a more resilient Internet infrastructure. Common practice in protocol design and network operations has largely been to blindly acceptance all protocol message exchanges among the network entities. However, given that faults and attacks are inevitable, we believe that this practice must be changed to incorporate additional verifications and validation steps. Cryptography-based protection mechanisms, such as DNSSEC and SBGP, are one of the steps in this direction, however their real deployment is slow in coming, and they themselves will suffer from break-ins, implementation defects, and more likely, human errors. This work explores a new venue in building protection mechanisms. Similar to the work reported in [17], it does not depend on cryptography but instead makes use of certain properties of the network infrastructure. Even after DNSSEC and other cryptography-based mechanisms become widely deployed, this type of approach will still provide an important added line of protection for DNS service. In a more general sense, any truly resilient system must include multiple protection fences since no single fence can be strong enough to defend against all potential faults and attacks.

9 Acknowledgments

We would like to thank the anonymous reviewers for their valuable comments, the RIPE RIS project for collecting the BGP log data, and Mark Kosters and a number of Verisign network administrators for providing information about gTLD server changes.

References

[1] R. Arends, M. Larson, D. Massey, and S. Rose. DNS security introduction and requirements. *Internet Draft*, Dec. 2002.

[2] D. Atkins and R. Austein. Threat analysis of the Domain Name System. *Internet Draft*, Nov. 2002.

[3] J. J. Garcia-Lunes-Aceves and S. Murthy. A loop-free path-finding algrithm: Specification, verification and complexity. In *Proceedings of the IEEE INFOCOM*, Apr. 1995.

[4] ICANN. DNS Security Update 1. http://www.icann.org/committees/security/dns-security-update-1.htm, Jan. 2002.

[5] S. Kent, C. Lynn, and K. Seo. Secure Border Gateway Protocol. *IEEE Journal of Selected Areas in Communications*, 18(4), Apr. 2000.

[6] C. Labovitz, A. Ahuja, A. Bose, and F. Jahanian. Delayed Internet routing convergence. In *Proceedings of ACM SIGCOMM*, August/September 2000.

[7] R. Mahajan, D. Wetherall, and T. Anderson. Understanding BGP misconfiguration. In *Proceedings of ACM SIGCOMM 2002*, Aug. 2002.

[8] D. McGuire. Getting to the root of all e-mail. *The Washington Post*, Mar. 2002.

[9] P. Mockapetris. Domain names–concept and facilities. *RFC 1034*, Nov. 1987.

[10] D. Pei, X. Zhao, L. Wang, D. Massey, A. Mankin, S. Wu, and L. Zhang. Improving BGP convergence through consistency assertions. In *Proceedings of the IEEE INFOCOM*, June 2002.

[11] Y. Rekhter and T. Li. Border Gateway Protocol 4. *RFC 1771*, July 1995.

[12] J. Rexford, J. Wang, Z. Xiao, and Y. Zhang. BGP routing stability of popular destinations. In *ACM SIGCOMM Internet Measurement Workshop 2002*, Nov. 2002.

[13] RIPE Routing Information Service. http://www.ripe.net/ripencc/pub-services/np/ris-index.html.

[14] B. Smith and J. J. Garcia-Luna-Aceves. Securing the Border Gateway Routing Protocol. In *Proceedings of Global Internet*, Nov. 1996.

[15] L. Wang, X. Zhao, D. Pei, R. Bush, D. Massey, A. Mankin, S. Wu, and L. Zhang. Observation and analysis of BGP behavior under stress. In *ACM SIGCOMM Internet Measurement Workshop 2002*, Nov. 2002.

[16] X. Zhao, D. Pei, L. Wang, D. Massey, A. Mankin, S. Wu, and L. Zhang. An analysis of BGP multiple origin AS (MOAS) conflicts. In *ACM SIGCOMM Internet Measurement Workshop 2001*, Nov. 2001.

[17] X. Zhao, D. Pei, L. Wang, D. Massey, A. Mankin, S. Wu, and L. Zhang. Detection of invalid routing annoucements in the Internet. In *The International Conference on Dependable Systems and Networks*, June 2002.