# Protecting DNS from Routing Attacks: A Comparison of Two Alternative Anycast Implementations

*Ioannis Avramopoulos*
*Deutsche Telekom Laboratories*
*Ernst-Reuter-Platz 7*
*10587 Berlin, Germany*
*ioannis.avramopoulos@telekom.de*

*Martin Suchara*
*Department of Computer Science*
*35 Olden Street*
*Princeton, NJ 08544*
*msuchara@princeton.edu*

## Abstract

DNS is a critical piece of the Internet supporting the majority of Internet applications. Because it is organized in a hierarchy, its correct operation is dependent on the availability of a small number of servers at the upper levels of the hierarchy. These *backbone* servers are vulnerable to routing attacks in which adversaries controlling part of the routing system try to hijack the server address space. Using routing attacks in this way, an adversary can compromise the Internet's availability and integrity at a global scale. In this article, we evaluate the relative resilience to routing attacks of two alternative anycast implementations of DNS, the first operating at the network layer and the second operating at the application layer. Our evaluation informs fundamental DNS design decisions and an important debate on the routing architecture of the Internet.

## 1 Introduction

The Domain Name System (DNS) is a critical piece of the Internet infrastructure providing a mapping between host names and IP addresses. This function is necessary because users identify endhosts using mnemonic character strings (e.g., `www.google.com`) whereas the routing system uses IP addresses (e.g., 140.13.12.1) to deliver the data between the endhosts.

To provide this function, DNS is organized in a hierarchical fashion—answering the majority of DNS queries requires information that is stored in a small number of servers that are close to the top of the hierarchy (we call them *backbone* servers)—the availability of those servers is essential for the correct operation of DNS and of the Internet as a whole. Because of the important role they serve in the Internet, these servers are attractive targets for attackers. They have been repeatedly attacked in the past typically using denial-of-service attacks in which compromised hosts make excessive DNS requests to prevent the servers from answering legitimate requests [1].

In this paper, we consider attacks against DNS that are more general than the aforementioned denial-of-service attacks. In the attacks that we consider the goal of the adversary is to either compromise availability (i.e., to prevent

the backbone servers from answering legitimate requests) or integrity (i.e., to provide a fake response to the DNS requests). To that end, we assume that the adversary is able to control one or more routers and uses the routing system to hijack the address space of the victim servers. Although no previous incidents are known of routing attacks against DNS, significant routing-based Internet disruptions have occurred in the past (e.g., when an ISP in Pakistan inadvertently hijacked YouTube's traffic [2]).

To improve their availability, the data served by the backbone DNS servers are typically replicated and client requests are *anycasted* to the replicas (meaning that each client request can be answered by any of the functionally equivalent replicas). In this paper, we investigate the relative strengths of two alternative anycast implementations, the first operating at the network layer (*network-layer anycast*) and the second operating at the application layer (*application-layer anycast*). Since both implementations are used in practice—e.g., the *root* DNS servers (at the topmost of the DNS hierarchy) are implemented using a combination of network-layer and application-layer anycast—one of the goals of our evaluation is to guide practical design decisions for the DNS system.

Using simulation experiments we find that network-layer and application-layer anycast can in principle both be effective in defending against routing attacks—their security performance (measured by the percentage of sources reaching a legitimate anycast server) asymptotically approaches 100% as more networks are enlisted in the service. However, although network-layer anycast is based on a simple implementation, to achieve the same security performance, application-layer anycast must be equipped with an oracle able to distinguish legitimate from adversarial routes—implementing this oracle is in general a hard problem. We have also been able to verify the conclusions of the simulations analytically.

We also compare network-layer and application-layer anycast from a performance perspective. We observe that application-layer anycast can adapt better to the service demand and achieve balanced load distribution by being flexible in taking into account application-layer metrics such as server load. However, we argue that network-layer anycast can also achieve a comparable level of flexibility. We conclude that network-layer anycast is an effective and practical defense against routing attacks targeting backbone DNS servers.

Previous work has considered path-filtering techniques for protecting backbone DNS servers from routing attacks [13]. Different from this work we consider anycast as the countermeasure. We should note that the contribution of this paper is not in the novelty of the security countermeasures that we evaluate, but in the relevance, we hope, of the discussion to the efforts on securing DNS, the routing system, and the interplay between the two. For example, our discussion can inform DNS design decisions on whether network-layer anycast, application-layer anycast, or a combination of the two is more preferable in practice. Furthermore, the conclusion that anycast is effective against routing attacks is useful in informing an important debate about the Internet routing architecture. General-purpose countermeasures against routing attacks, such as secure routing protocols [10], have been notoriously hard to deploy and it is unlikely that they will be deployed in the near future. Fortunately, solving the important problem of protecting

the DNS backbone from routing attacks does not depend on deploying secure routing protocols.

## 2 DNS and Interdomain Routing

Before discussing DNS threats and countermeasures, we provide in this section background on DNS, an application-layer service that relies on the network layer to connect clients to DNS servers. We also provide background on the interdomain routing system that establishes the network-layer paths used by DNS.

### 2.1 DNS

DNS is a distributed database providing a mapping between host names and IP addresses and a protocol for making queries to this database. The names assigned by DNS to Internet hosts, called *domain names*, follow a hierarchical structure. A domain name is a sequence of character strings separated by "." (e.g., `www.princeton.edu`)—proceeding from right to left in a domain name is like traversing a path in a tree-based hierarchy. At the topmost level of the hierarchy is the root domain which corresponds to the empty string. At the next level are the top-level domains such as `com` and `edu` followed by second level domains such as `google.com` and `princeton.edu`. `www.princeton.edu` is an example of a host name that corresponds to the web server of Princeton University.

Following the hierarchical organization of domain names, the DNS database is stored in a hierarchy of nameservers that is each responsible for answering queries regarding a corresponding zone. Each zone contains information about a subset of the DNS namespace (including mappings of names to IP addresses and pointers to other nameservers). Nameservers can be classified into three types: root, top-level, and authoritative. A host that seeks to resolve a query (called *resolver*) typically proceeds in a sequence of steps in an iterative fashion contacting a new nameserver at each step—each nameserver provides additional information to the resolver until a nameserver that is able to answer the query is found. First, the resolver queries a root nameserver. Supposing the query is for `www.princeton.edu`, the root nameserver points the resolver to a top-level nameserver responsible for the `edu` zone. Similarly, this nameserver points the resolver to an authoritative nameserver responsible for the `princeton.edu` zone who is finally able to authoritatively answer the resolver's query (i.e., provide the IP address of host `www.princeton.edu`). We will refer to the root and top-level nameservers as *backbone* nameservers, because of their position in the DNS hierarchy.

The information in the DNS database is distributed across the Internet—each administrative entity managing a network may also manage its own portion of the DNS database stored in corresponding authoritative nameservers. However, unlike the authoritative nameservers that are responsible for answering queries about a small portion of the database, the root and top-level nameservers (operated by entities such as Verisign [3]) must in principle be contacted for every DNS query made in the Internet. Although caching responses obviates the need to contact the backbone nameservers on every query, the cache records have a limited lifetime and must eventually be refreshed. Therefore, backbone nameservers are attractive targets for attackers—failure of these servers implies a global-scale disruption of

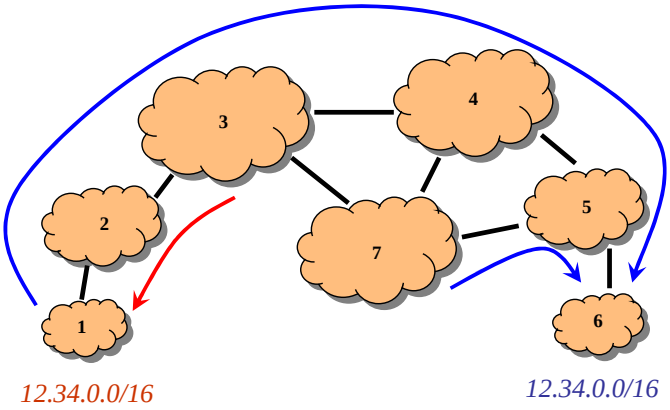


Figure 1: Announcement of prefix 12.34.0.0/16.

the Internet.

## 2.2 Interdomain Routing

Since DNS is an application-layer protocol, it relies on the network layer to provide connectivity between resolvers and nameservers. The protocols responsible for establishing the paths along which DNS traffic is forwarded are called *routing protocols*. The routers executing these protocols belong to independent network domains called *autonomous systems* (ASes). Routing within an AS is administered by *interior* routing protocols and routing across ASes is administered by the Internet's *exterior* or *interdomain* routing protocol, the Border Gateway Protocol (BGP) [12]. The root and top-level nameservers are typically accessible through BGP paths.

BGP establishes reachability to blocks of IP addresses called *prefixes* announced (or *originated*) by ASes. A prefix is denoted by an address followed by a forward slash and a decimal number showing the length of the prefix. BGP, to a first approximation, operates at AS-granularity. It is a path vector protocol meaning that a BGP announcement contains the sequence of ASes that must be traversed en route to the destination. For example, in Figure 1, suppose that AS6 announces prefix 12.34.0.0/16 to AS5. Then, AS5 announces to AS4 and AS7 that it can reach prefix 12.34.0.0/16 via AS6. This process continues until each AS receives a path toward AS6.

BGP selects routes according to AS-specific routing policies. These policies depend on multiple objectives including performance (as measured by the AS hop count) and the business relationships with neighboring networks. For example, a revenue-generating route to a customer is more preferable than a costly route to a provider. Finally, in BGP each router selects exactly one path to forward traffic toward the destination.

# 3 Threat Model

A BGP-speaking router trusts that its neighbors propagate legitimate information. This assumption of trust can be exploited by adversaries, who may use BGP attacks to hijack DNS traffic.

In BGP each prefix is usually announced by a single AS. However, this condition may be violated in practice either for legitimate reasons (such as the announcement of an *anycast prefix* as discussed later) or as part of a malicious attack. In the latter case, the adversary receives victim traffic, and may either act as a sink and discard the traffic, or try to impersonate the legitimate destination. The term *prefix hijacking* usually refers to the announcement of a victim prefix by an adversary. An AS that selects an adversarial route will propagate it to its neighbors, who may select it as well. For example, in Figure 1 prefix 12.34.0.0/16 is initially announced by AS6, and all sources point their routing tables toward AS6. If AS1 also announces this prefix, the network is partitioned in two subsets of ASes according to the origin AS they have chosen: AS2 and AS3 point their routing tables toward AS1, whereas AS4, AS5, and AS7 point their routing tables toward AS6.

In another variant of hijacking, the adversary breaks (or *deaggregates*) the victim's prefix into multiple sub-prefixes and originates those instead. In this way, although the network will maintain routes to both the original prefix and the sub-prefixes, because the data plane forwards traffic toward more specific prefixes, all traffic will be directed to the adverary-controlled sub-prefixes. We refer to this attack as *subprefix hijacking*. However, there is a limit to the granularity that the adversary can deaggregate a prefix. This limit is imposed by filtering rules employed by ISP networks that discard routes to prefixes more specific than /24. Therefore, if a prefix is a /24 prefix, it is not vulnerable to subprefix hijacking attacks.

There are still other variants of hijacking, such as *wormhole attacks*. Wormhole attacks are a countermeasure that an adversary can employ against secure routing protocols. Secure routing protocols are meant to prevent direct prefix hijacking attacks. However, they are vulnerable to more sophisticated attacks in which two colluding ASes announce a forged link so that routes containing this link are shorter and more attractive than legitimate routes.

Using the aforementioned attacks, the adversary can hijack the traffic destined to the root and top-level nameservers. Once the adversary receives the DNS traffic, he may simply discard it to affect *availability* by disconnecting the nameservers from the resolvers. Alternatively, the adversary can try to impersonate the nameservers to compromise *integrity* by providing false responses. Finally, instead of targeting the backbone servers, the adversary may also attack authoritative DNS servers further down in the DNS hierarchy. These latter attacks are of limited scope by comparison to attacks against the backbone servers and are not considered in this paper. Note that because authoritative servers are typically not distributed across wide area networks, their availability is closely dependent on the availability of the regular data traffic. Therefore, the protection of these servers is more likely to require more general countermeasures against routing attacks than the ones discussed in this paper.

## 4 Defense against Routing Attacks

The backbone DNS servers are organized in groups based on the zone which they serve. (E.g., the group of *root servers* serve the *root zone*.) The servers in each group are able to provide equivalent answers to DNS queries about

their respective zone. Therefore, it is sufficient to direct DNS queries to any of a group's members. This process is called *anycast*. Anycast implementations are classified according to the layer in which they operate (i.e., network layer vs. application layer).

After discussing network-layer and application-layer anycast in greater detail, we compare their relative resilience to routing attacks. First, we find that against a network-layer anycast implementation, the strength of the adversary diminishes as the anycast group size increases. Second, we find that the security performance of network-layer anycast is very close to the security performance of ideal application-layer anycast that uses an oracle able to distinguish legitimate from adversarial routes.

## 4.1 Network-layer Anycast

In network-layer anycast [11], two or more endhosts (called anycast *targets*) are assigned the same IP address and the routing system is responsible for forwarding traffic destined to this address to *anyone* of the targets. Interdomain network-layer anycast is typically implemented through the simultaneous origination of an IP prefix (called an *anycast prefix*) from multiple autonomous systems in which case BGP is responsible for determining which target receives the traffic. From the perspective of BGP, an anycast IP address is like any other regular IP address (except that it cannot be aggregated). Operationally, announcing an anycast prefix from multiple ASes simply results in a possible increase in the number of alternative routes received by a BGP router and the normal BGP route selection process determines which one of those routes to accept for forwarding.

The primary function of anycast in existing deployments is to perform load balancing across a number of servers. From a security standpoint, network-layer anycast has been proven effective in defending against denial-of-service attacks by exactly balancing the offending traffic across the anycast targets [1]. In this article, different from a load balancing problem, we are interested in whether anycast can be effective in preventing an adversary from hijacking the anycast traffic.

And here comes our key observation: In order for a server to become an anycast target, its corresponding autonomous system must announce the anycast prefix in BGP. From an operational perspective, this is identical to the aforementioned prefix hijacking attack—an adversary controlling an autonomous system would perform the same steps to hijack the anycast traffic. Therefore, in exactly the same manner the adversary is successful in retrieving the victim traffic through prefix hijacking, the anycast group can also be successful in taking back that traffic through the deployment of additional autonomous systems participating in the group. Essentially using network layer anycast to replicate the backbone DNS servers, the network operators manage to *hijack the hijacker*.

## 4.2 Application-layer Anycast

In a service implemented using application-layer anycast [14], a client must engage in an application-layer interaction to obtain one or more IP addresses of corresponding servers that we will also refer to as anycast *targets*. Different from network-layer anycast in which all targets share the same IP address, in application-layer anycast each server has a separate address. It is then the responsibility of the client to contact one of the targets. In the existing implementations of application-layer anycast, the interaction between clients and service is happening through DNS itself. In such implementations, the service is typically identified with a domain name and the authoritative servers for this domain name are configured to respond to clients with a custom list of IP addresses for the service. Then each client becomes responsible for selecting one of these addresses to obtain the service—typically a client selects the first IP address in the list although more flexible selection processes are also possible.

Although such an implementation of application-layer anycast is applicable to the majority of anycast traffic, it does not apply to the selection of root DNS servers—because these servers are used to bootstrap DNS, it is not possible to use DNS itself to map clients to the root servers. The root servers are accessible instead using an alternate application-layer implementation—the network authority that is responsible for managing the root servers (the Internet Assigned Numbers Authority) regularly publishes a file containing the full list of root IP addresses (13 addresses in total). Network administrators are responsible for downloading this file from a well-known network location and configuring their resolvers to issue queries to a particular address in the list.

Before evaluating the security performance of the two alternative anycast implementations we note that in attacking application-layer anycast, the strategy of the adversary must differ slightly from the one used against network-layer anycast. Against network-layer anycast, the adversary simply announces the anycast prefix. In application-layer anycast, because each participating AS announces a separate prefix, we are going to assume that the adversary announces each one of those prefixes. We also assume that all prefixes are /24 prefixes so that subprefix hijacking attacks do not apply.

## 4.3 Evaluation of Network-Layer Anycast

In this section, we evaluate the security performance of network-layer anycast using simulation. Our experiments simulate the propagation of BGP route announcements on the AS-level Internet topology and how the routing tables evolve. Route propagation is influenced by AS business relationships as discussed earlier. To accurately model the route propagation, we used a CAIDA AS topology [4] annotated with inferred business relationships. Constructed by analyzing routing table snapshots of RouteViews servers [5], this dataset is one of the most complete and accurate. We simulated policy-based route propagation on this topology using BSIM [6]. BSIM accurately captures the influence of business relationships on how ASes select and export routes. Routing table inspection at the end of each experiment allows us to determine the fraction of ASes with valid routes to the victim prefix.

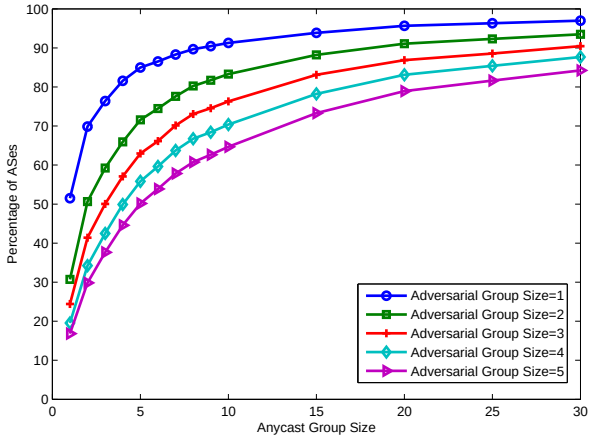


Figure 2: Security performance of network-layer anycast.

To measure the impact of an attack, we calculate the average fraction of ASes that accept a route to a legitimate anycast target over a sequence of 100 experiments, where in each experiment

1. we assume that the members of the adversarial group perform a prefix hijacking attack against the anycast prefix. In our setting, this is the strongest possible attack the adversary can perform.

2. we select the members of the anycast and adversarial groups at random. In this way, we are able to estimate how difficult it is for the adversary to mount a successful attack—as the average performance increases the number of locations the adversary is able to perform a successful attack decreases.

Figure 2 shows the percentage of ASes accepting a route toward a legitimate anycast target as a function of the size of the anycast group. In the figure, there are five plots corresponding to adversarial groups of size one to five. As shown in the figure, security performance depends on both the size of anycast group (say $x$) and the size of the adversarial group (say $y$), being roughly $x/(x+y)$. Therefore, using network-layer anycast, the defending entity can control the security performance by increasing the number of ASes advertising the anycast prefix. As the size of the anycast group increases, the power of the adversary diminishes.

## 4.4 Evaluation of Application-Layer Anycast

In network-layer anycast, all participating autonomous systems advertise the same anycast prefix, and each client accesses the anycast service through exactly one route chosen by the routing system. In contrast, in application-layer anycast each participating AS advertises a separate prefix. In this way, each client is able to choose one of several possible routes toward the service by using a corresponding IP address as the destination address. Therefore, in application-layer anycast, clients have more choice. However, more choice by the clients does not result in better security as the following simulation experiments illustrate.

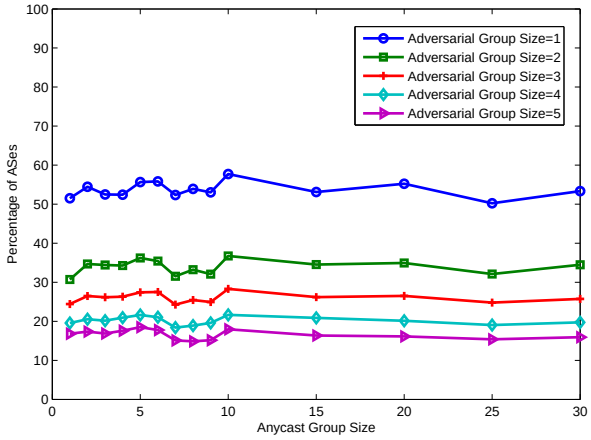


Figure 3: Security performance of naive application-layer anycast.

We use the same simulation environment as before, and we assume each adversarial autonomous system performs a prefix hijacking attack against all prefixes advertised by the anycast group members. We evaluate two alternative application-layer anycast implementations. In the first, each source selects the anycast target at random. This rule for selecting an anycast target is the one most frequently encountered in application-layer anycast in the Internet: As mentioned earlier, in application-layer anycast, clients typically receive a list of IP addresses from an authoritative DNS server and direct the service request toward the first IP address in the list. Before responding to a query, the authoritative DNS server permutes the list of IP addresses so that the assignment of servers to clients is performed in a round-robin fashion. Such an assignment is essentially equivalent to a random assignment. Figure 3 shows the percentage of autonomous systems accepting a route toward a legitimate anycast target as a function of the size of the anycast group. Note that the security performance is very poor—it is roughly $1/(1+y)$, where $y$ is the size of the adversarial group, irrespective of the size of the anycast group.

In the second implementation, the sources select the anycast target using an oracle that, given a route, is able to determine whether the route leads to a legitimate target or the adversary. Using this oracle a source selects a route to a legitimate target as long as such a route is available at the source. The security performance attainable using this implementation is very close to the security performance of network-layer anycast shown in Figure 2—the difference being at most 0.1% at any given experimental configuration.

The previous experiments illustrate two main points. First, as the first experiment shows, application-layer anycast may lead to poor security performance, significantly worse than the security performance of network-layer anycast. Second, as the second experiment shows, even ideal application-layer anycast does not provide better security than network-layer anycast. In fact, attaining the security performance of ideal application-layer anycast requires implementing an oracle that is able to distinguish adversarial from legitimate routes, which is generally a hard problem.

We have been able to verify the conclusions of the simulations in this and the previous subsection analytically. It is interesting to note that the security performance of network-layer and application-layer anycast is close under weak assumptions about the routing system. (The proofs are available upon request and will appear in a subsequent technical report.)

## 5 Achieving Balanced Load Distribution

The implementation of the service that provides access to backbone DNS zones must not only be resilient to routing attacks but also provide good load distribution properties that are important not only during regular operations but also during denial-of-service attacks.

As discussed earlier, the primary function of anycast in existing deployments is to assign clients to servers so that each client is assigned to a close-by server and the load across the servers is balanced. However, in network-layer anycast, this assignment is made by the routing protocols whereas, in application-layer anycast, the assignment is made by the application itself. The designer of an anycast application is therefore presented with a dilemma: in network-layer anycast, the routing protocols do not learn about application-specific metrics whereas, in application-layer anycast, applications have little control over network paths. Controlling network paths was important to achieve good security performance in the previous section. In this section we argue that network-layer anycast can also achieve good load distribution.

With respect to assigning DNS clients to servers, application-layer anycast has a comparative advantage over network-layer anycast. In application-layer anycast, clients receive the list of server IP addresses and are able to choose which IP address to direct their request to. In their selection of the "best" server, clients are able to take into account both network-layer and application-layer performance (albeit in a crude fashion), by measuring the response time corresponding to each server selection (whether passively or actively using probe packets). The response time of each server selection is the sum of the network and server response times and, therefore, takes into account a combination of both network performance and server performance. Thus application-layer anycast is flexible in approximately achieving both server selection according to proximity and load distribution across the servers.

In selecting paths for an application implemented using network-layer anycast, the routing system takes into account performance metrics (implicitly through the network operators performing traffic engineering). Although using traffic engineering (and other related techniques such as *overprovisioning*) the routing system is generally able to select good-performing paths, the routing system is not informed about the load on the individual DNS servers and therefore cannot make routing decisions according to server load. However, an application implemented using network-layer anycast can compensate for this disadvantage differently: It is possible to control the load on each application server, first, by adjusting how many and which ASes participate in the anycast group, and, second, by adjusting the number of application servers in each participating AS. In this way, the load on each of the application servers can be balanced

according to application-specific requirements.

Network-layer anycast has proven its worth with respect to its load distribution properties in practice during the denial-of-service attack mounted against the root DNS servers in early 2007 [1]. In this attack, compromised hosts bombarded 6 of the 13 root-server IP addresses with excessive traffic. Two of the victim addresses were unicast addresses and the corresponding servers were severely affected. The remaining 4 victim addresses were anycast addresses and the corresponding servers were able to successfully withstand the attack.

From the perspective of defending against denial-of-service attacks, network-layer anycast is more preferable than application-layer anycast: In network-layer anycast, the assignment of traffic load to the servers is under the control of the routing system and the service operators, whereas in application-layer anycast it is the adversary that determines the distribution of the traffic load to the servers. Therefore, if a service is implemented using network-layer anycast, the network and service operators can act individually and in concert to contain the damage by changing the routing and controlling the service replication. In contrast, in application-layer anycast, the adversary can more easily adapt to such countermeasures.

## 6 Comparison with Other Proposals

In being an effective defense against routing attacks, anycast can complement or even substitute for other proposals for securing the Internet. In this section, we discuss how anycast compares to DNSSEC and secure routing protocols.

### 6.1 Comparison with DNSSEC

DNSSEC [9] is a security extension of the DNS protocol aiming to protect the authenticity and integrity of DNS data. DNSSEC does not aim to protect the availability of DNS data, which is a first-order goal of the techniques presented in this paper. Therefore, DNSSEC and anycast can act in a complementary fashion.

Using a prefix hijacking attack, an adversary can compromise the authenticity and integrity of DNS data by capturing the queries from DNS resolvers and providing false responses to those queries. The backbone DNS servers are attractive targets because they serve queries for most of the Internet traffic. Thus, by limiting the amount of backbone DNS traffic the adversary is able to hijack, anycast also protects the authenticity and integrity of DNS data. It is worth noting that since DNSSEC has received limited deployment traction, anycast may substitute for DNSSEC at least in the early stages of DNSSEC deployment.

### 6.2 Comparison with Secure Routing Protocols

Several techniques have been previously proposed to secure routing in the Internet but none has received deployment traction. The most notable proposal is Secure-BGP [10]. Secure-BGP requires from all ASes (approximately 30,000) to participate in secure registries and to cryptographically validate BGP messages—supporting such functionality requires costly infrastructure upgrades. Therefore, adoption of Secure-BGP requires that ASes make a costly investment.

However, the incentives in this very large group of heterogeneous entities (ranging from large backbone providers to small enterprise networks) are not necessarily aligned to act jointly in such an endeavour—the benefits and costs of deployment vary widely across ASes. Although it is certainly true that any network would rather avoid having its traffic hijacked by an adversary, the protection benefits of an adopter must outweigh the costs. This has proven hard to achieve especially at the early stages of deployment when the protection benefits are low.

In contrast to the deployment difficulties of secure routing protocols, protecting the backbone DNS servers from routing attacks using anycast requires only unilateral action by the corresponding management entities. The end users and the authoritative servers are not required to make any effort to receive the protection benefits. Therefore, anycast is easily deployable. Although the protection of the DNS traffic is narrower than the general-purpose protection provided by secure routing protocols, it is possible to extrapolate the techniques used to protect the DNS traffic along two dimensions. First, a more secure DNS system can serve as an ingredient of a more complete solution protecting the data traffic itself. Second, anycast can independently serve as a stand-alone ingredient in systems that secure the unicast data traffic. Discussing such a mechanism is beyond the scope of this paper, and we refer the reader to our technical report [7].

Finally, note that in comparing secure routing protocols and anycast, it would seem that in full deployment secure routing protocols could provide better protection for the backbone DNS servers than anycast. However, this is not generally true—even if a secure routing protocol is ubiquitously deployed, it is possible for the adversary to hijack the victim DNS traffic using the aforementioned wormhole attacks. Therefore, anycast and secure routing protocols can, in principle, even act in a complementary manner in defending against routing attacks.

## 7 Conclusion

Backbone DNS servers are vulnerable to routing attacks in which an adversary controlling one or more routers advertises the victim IP addresses in BGP. Using the routing system in this way, by affecting the operation of a small number of servers, the adversary can mount global scale attacks. In this article, we have investigated the relative strengths of network-layer and application-layer anycast in defending against routing attacks, and we have found that, although both can achieve comparable security, network-layer anycast can be effective using a simple and practical implementation. Therefore, anycast obviates the need to deploy heavyweight countermeasures such as secure routing protocols in defense of the DNS backbone servers.

Beyond evaluating the two basic anycast designs considered in this article, we believe that our work is a first step in understanding the resilience to routing attacks of alternative anycast designs such as the one proposed in [8]. We also believe that anycast can serve as a basis for more general countermeasures against routing attacks targeting the regular data traffic. We leave the investigation of such general-purpose countermeasures as future work.

## References


[1] *http://blogs.zdnet.com/security/?p=118*.

[2] *http://www.ripe.net/news/study-youtube-hijacking.html*.

[3] *http://www.root-servers.org/*.

[4] *http://as-rank.caida.org/data/*.

[5] *http://www.routeviews.org/*.

[6] *http://www.cs.unm.edu/~karlinjf/pgbgp/*.

[7] I. Avramopoulos, M. Suchara, and J. Rexford. How small groups can secure interdomain routing. Technical Report TR-808-07, Princeton University Computer Science Department, December 2007.

[8] H. Ballani and P. Francis. Towards a global IP anycast service. In *Proc. ACM SIGCOMM*, Aug. 2005.

[9] R. Chandramouli and S. Rose. Challenges in securing the Domain Name System. *IEEE Security and Privacy*, 4(1), Jan.-Feb. 2006.

[10] S. Kent, C. Lynn, and K. Seo. Secure Border Gateway Protocol (Secure-BGP). *IEEE Journal on Selected Areas in Communications*, 18(4):582–592, Apr. 2000.

[11] C. Partridge, T. Mendez, and W. Milliken. Host anycasting service. RFC 1546, IETF, Nov. 1993.

[12] Y. Rekhter, T. Li, and S. Hares. A border gateway protocol 4 (BGP-4). RFC 4271, IETF, Jan. 2006.

[13] L. Wang, X. Zhao, D. Pei, R. Bush, D. Massey, and L. Zhang. Protecting BGP routes to top-level DNS servers. *IEEE Transactions on Parallel and Distributed Systems*, 14(9), Sept. 2003.

[14] E. Zegura, M. Ammar, Z. Fei, and S. Bhattacharjee. Application-layer anycasting: A server selection architecture and use in a replicated web service. *IEEE/ACM Transactions on Networking*, 8(4), 2000.