Incentives and Disincentives for DNSSEC Deployment

Christof Fetzer and Trevor Jim

March 1, 2004

Abstract

Like many of the Internet's foundational infrastructures, the Domain Name System (DNS) suffers from a number of security vulnerabilities. Efforts to produce a more secure successor, DNSSEC, have been underway for more than ten years, and a specification is only now reaching a final form; this is an eternity in Internet time. We argue that the reasons for this delay are mainly economic rather than technical, and that these reasons will persist even after the specification is finalized and robust implementations are available. Thus, widespread adoption of DNSSEC is by no means certain. We offer several suggestions to increase DNSSEC's odds of success.

1 Introduction

The Domain Name System, or DNS, is the distributed database that maps hostnames to the numeric IP addresses needed to route communications between hosts. It is a fundamental service of the Internet, used each time a person clicks on a link in their browser to bring up a new web page. Like most other fundamental services of the Internet, DNS was designed in an era when security was less of a pressing issue, and it is vulnerable to a variety of attacks, such as denial of service [3] and cache poisoning [1], to name just two.

Efforts to make DNS more secure have been underway for a long time. One effort in particular is an extension of DNS called DNSSEC. DNSSEC has been under discussion since at least 1993, but the DNSSEC specification is only now approaching a final state (March 2004), and it is not yet deployed. This is quite a long time: ten years. Technical hurdles and the usual contentions of a standards process explain part of the delay, but we believe that other factors, including economic factors, have played a more important role. Moreover, these economic factors will not disappear after the specification is finalized, because the specification does not address them. Unless addressed, they will hinder DNSSEC's adoption.

2 DNS and DNSSEC

DNS is a service that maps hostnames (like example.com) to IP addresses (like 192.0.2.1) as well as other data. This data is not stored on just one computer, but rather is distributed among many *nameservers* throughout the network. When a nameserver is queried for a host's address, it can either return the address (if it has it), or indicate another nameserver that might be able to answer the query. Obtaining an address thus may involve querying several nameservers; a program called a *resolver* performs this task for clients.

The communication between resolvers and nameservers is insecure: an adversary can monitor queries from a resolver and substitute its own responses for the responses of the nameservers. In this way, the resolver may be fooled into believing that a host has any IP address the adversary chooses.

DNSSEC is designed to foil this sort of attack. It specifies how to sign DNS data with cryptographic keys, and how to store the public keys and signatures in DNS itself. It also specifies how resolvers can retrieve keys and signatures, and how they can verify signatures on the DNS data. The end result for clients is that when they query DNS for the address of a host, they are assured that the address returned has been supplied by authorized nameservers, and not by an adversary.

Numerous technical difficulties had to be overcome during the development of DNSSEC. The thorniest issues, however, had to do with interoperability—making sure that DNSSEC-aware resolvers and nameservers work with DNSSEC-ignorant resolvers and nameservers. These issues are well within the capabilities of the DNS experts involved in the IETF process; in the end, no fundamental breakthroughs or novel security mechanisms were needed. Technical factors do not begin to explain DNSSEC's over-long development. Instead, we must look to other factors, including economic factors.

First, we must note that *by itself*, DNSSEC provides little of value. Discovering the correct IP address of a host does not guarantee that the routing infrastructure has not been corrupted, or that an adversary is not intercepting and modifying your communications with that address. DNS is only one step in the network communication, and replacing DNS with DNSSEC alone will not secure the communication.

Furthermore, we have no evidence that address spoofing is anything but exceedingly rare, despite longstanding published reports of DNS vulnerabilities [1, 10, 6, 4, 5]. (There have been distributed denial of service attacks on the DNS root nameservers [11], but DNSSEC does not by itself prevent denial of service.) It seems that no one is trying to corrupt the addresses of the great majority of Internet hosts. Using DNSSEC to protect them may be like a bakery using an armored car to deliver rolls: the rolls certainly have value, but not that much value. In other words, it is economically inefficient (the costs exceed the benefits).

The value added by DNSSEC in isolation therefore may not justify a large investment, and, indeed, little in the way of resources has been expended on DNSSEC. There has been no consumer demand or awareness of DNSSEC, and there has been little industry push or contribution. DNSSEC has moved forward based on the efforts of a relatively small number of people involved in the IETF. DNSSEC has not been a priority and has received far less effort than other, more pressing issues, such as spam.

This is not to say that DNSSEC has no value; it does, particularly in combination with other efforts (which we discuss in the next section). And there are situations such as online banking where the kind of validation provided by DNSSEC is not only useful, but essential (customers demand additional security). However, in these cases DNSSEC faces another economic challenge: competition.

3 The competition

The Secure Sockets Layer, or SSL, is a protocol for authenticated and encrypted network communications. It relies on public key certificates signed by trusted third parties called certificate authorities (CAs). An SSL certificate asserts that a given public key is associated with a hostname. In a typical scenario, a client looks up the address of a hostname in DNS, contacts the address, receives a certificate from the server, and checks that the hostname of the certificate matches the desired hostname, that the signing CA is trusted, and that the signature is valid. The server's public key from the certificate can then be used to authenticate and encrypt further communications.

At first glance, it may not appear that SSL competes with DNSSEC, since SSL does not ensure the integrity of DNS data. Note however that SSL does provide some validation for the address obtained from (standard, insecure)

DNS for a given host. Namely, SSL tells the client that a trusted CA thinks that the host reached using the address is in fact associated with the hostname. This is not exactly the same as what DNSSEC provides, but the average user (the consumer) does not care about the difference.

Moreover, unlike DNSSEC, SSL provides authentication and encryption, which are required for applications like online banking—in other words, exactly the sorts of applications where DNSSEC would provide the most value. And SSL has other advantages over DNSSEC:

- SSL is mature. It has been deployed since 1994, and the protocol and implementations have been examined extensively. This has resulted in several revisions and the protocol has gone through the standards process. (The final standard is called Transport Layer Security, or TLS.)
- SSL has brand awareness. Consumers know when they are using SSL (typically, because of an "https" prefix in their browser, and/or a visible key or lock icon), and they demand it for financial transactions conducted over the web.
- SSL has a business model. There is a thriving industry of Certificate Authorities who sell certificates, and a rapidly growing new industry of SSL-based VPN providers.
- SSL is entrenched. It is supported by all major operating systems and web browsers and requires little or no further development or investment.

In short, SSL is a formidable competitor to DNSSEC. By taking over the high-value end of the market, it has reduced the need for DNSSEC and arguably diverted resources that might have helped develop DNSSEC faster.

Other DNSSEC competitors include the many public key infrastructure (PKI) efforts, for example, XKMS [7]. Although these PKIs do not seek to secure DNS data, they are competitors in the same sense as SSL: they provide additional services that consumers demonstrably value, and they do not require DNSSEC to secure IP addresses. Instead of DNSSEC, they rely on the existing DNS to obtain network addresses, but, like SSL, they perform additional checks to make sure that the correct party is contacted. This is a "trust, but verify" strategy.

"Trust, but verify" is a standard technique; in fact, it is used by DNSSEC itself. When a DNSSEC nameserver answers a query by indicating another nameserver to query, it may return unsigned "glue records" which a resolver uses to contact the next nameserver; the data in the glue records is verified by the resolver separately. So with respect to obtaining the correct IP address, DNSSEC and its competitors differ most markedly not in the strategy employed, but rather in the parties trusted (the trust model).

Advantages of DNSSEC. In spite of any criticism we might make of DNSSEC's development and eventual deployment, it must be admitted that it does solve a hard problem. As many have noted, DNSSEC is essentially a special-purpose PKI; having already solved some of the difficult problems inherent in PKIs, it could be extended to support a general PKI fairly easily (e.g., see [8]). We are not going to argue that this path should be pursued, because it is technically controversial and because the economics of such an approach are uncertain.

On the other hand, we have already argued that DNSSEC by itself is not economically attractive; some additional value must be extracted from the DNSSEC infrastructure to make it worth deploying. At various points in the history of DNSSEC, it has supported additional features such as distributing cryptographic application keys, but these features were stripped out of DNSSEC to simplify the

specification (and hence, speed finalization of the spec). We believe that to make DNSSEC deployment economically feasible, some of these features must be deployed along with DNSSEC, not as part of the spec, but rather as part of a bundling strategy.

The most promising of the additions would be IPSECKEY. This is a proposal for a DNS resource record to hold cryptographic keys for IPSec. If DNSSEC were deployed along with IPSECKEY and appropriate hooks in an IPSec implementation, then DNSSEC would be an integral part of a VPN solution and could thus compete on stronger terms with, for example, SSL VPNs.

In this application, DNSSEC has several advantages over its competitors.

- IPSec key distribution potentially involves keys for every host on the Internet. DNS has been designed for this large scale and experience has shown that it can handle the required traffic volume. It is a lightweight protocol that relies mostly on UDP, and, although packets will be larger for DNSSEC, it appears that UDP will still be workable. The most heavily loaded nameservers, the roots and the TLD nameservers, are replicated and geographically distributed, and have enough excess capacity to weather a high volume of unnecessary traffic generated by poorly programmed and/or misconfigured resolvers [12, 13] and even a distributed denial of service attack [11].

- IPSec, in contrast to SSL VPNs, provides security below the application layer, meaning that it protects legacy applications. It also protects UDP traffic as well as TCP traffic, while SSL protects only TCP traffic. SSL VPNs also typically work through a browser interface, so other TCP applications are not supported. An IPSec/DNSSEC combination would thus offer VPN service to many more applications.

- DNS is the right place to keep information maintained for all hosts, because the trust model of DNS follows exactly the administrative structure of the network. A single trust model is desirable to reduce inconsistencies.

- DNS is very widely deployed, and BIND [2], the most widely used implementation of DNS, has been extended to support DNSSEC. We can expect that DNSSEC will gradually become widely available (even if not used) as organizations upgrade BIND. Systems administrators are already familiar with BIND and so face a reduced learning curve for DNSSEC. This is not so much an advantage over SSL (which has similar deployment), but it is an advantage over the other PKI competitors.

4 Vested interests

A key player in the future of DNSSEC will be VeriSign, which maintains the .com generic top-level domain (gTLD) of DNS through its subsidiary, Network Solutions.[1] The .com gTLD is the largest and most valuable of the gTLDs, by far, and, hence, the support of VeriSign is crucial for the success of DNSSEC. It is by no means certain that VeriSign will agree to transition to DNSSEC. There are likely to be two sticking points: cost and SSL.

Cost. DNSSEC does not come for free.

- Bandwidth and storage requirements are increased by about a factor of 6 over DNS, a significant factor for a large gTLD like .com; the .com zone file would increase from about 4GB to 24GB. VeriSign's current DNS implementation keeps the zone

[1] As of October 2003 VeriSign has announced that they will sell Network Solutions but will maintain a minority interest of 15%.

file in RAM, hence this might require a significant hardware upgrade.

- Signing a zone is computationally expensive and needs to be repeated when the zone changes, and a zone like .com changes frequently.

- DNSSEC imposes new administrative costs. Keys must be managed: they must be created and protected. If a key is compromised it must be changed, and it is good practice to change keys from time to time anyway. Key changes may require coordination between a domain and a subdomain.

- In 2002 through 2003, VeriSign shifted their DNS implementation from BIND to a propriety solution called ATLAS, developed in-house at a claimed cost in the millions of dollars US. The existing DNSSEC implementations are versions of BIND, not ATLAS, and it is unlikely that VeriSign will transition back to BIND. So, there would be additional testing and development costs for VeriSign to support DNSSEC through ATLAS.

Furthermore, while DNSSEC is designed to allow partial and gradual adoption, so that .com can be secured with DNSSEC before a subzone like example.com, the overhead of DNSSEC at a zone applies regardless of whether subzones are secured. This means that VeriSign would pay the full costs of DNSSEC regardless of how many subzones migrated to DNSSEC. (A proposal to design DNSSEC in such a way as to avoid this problem was defeated in the standards process.)

Should VeriSign decide to deploy DNSSEC, it is unlikely to do so without passing these costs on to customers. Since a secure entry in DNS seems more valuable than an insecure entry, VeriSign will want to charge more for secure entries, and this differential price will deter DNSSEC use.

SSL. As we argued in section 3, DNSSEC and SSL are competing technologies. DNSSEC has the potential to undermine SSL and its certificate authorities, for example, by promoting IPSec VPNs over SSL VPNs.

This should concern VeriSign, because in addition to operating the .com gTLD, they are the largest of the SSL certificate authorities. VeriSign is unlikely to support a technology that could reduce their income from their SSL business, unless they can obtain equal or greater income by shifting to the new technology. Among other things, this implies that, should VeriSign support DNSSEC, they would like to charge a similar price for a secure DNSSEC entry as for an SSL certificate. Currently, this is about $1,000 per certificate per year.

5 A strategy for deployment

We now summarize the economic factors affecting DNSSEC and use them to suggest a strategy for effectively deploying the technology.

1. DNSSEC does not add enough value by itself to justify the costs of deployment.

2. The competition is SSL.

3. A combination of DNSSEC combined with support for distributing IPSec keys would provide a valuable service with competitive advantages.

4. VeriSign is unlikely to deploy DNSSEC without substantial compensation.

Items (1–3) imply that DNSSEC should be deployed as a bundle with an IPSec key distribution mechanism such as IPSECKEY, and with an IPSec implementation with built-in support for DNSSEC and IPSECKEY.

Of course, in suggesting this we are aware that many people have been working towards just such an integration of DNSSEC and IPSec. However, it is also true that there have been

many other similar proposals (distributing SSH keys in DNSSEC, building general PKIs on DNSSEC, etc.) and that to a certain extent each of these proposals has been pushed forward separately. We are emphasizing that it is essential to pick the combination of proposals offering the most value for the effort and push them forward as a package, ignoring the other proposals for expediency. We predict that a strategy that pushes DNSSEC alone will not succeed.

This brings us to item (4). The conventional wisdom for DNSSEC deployment has been that first, the root zone will be secured, as it seems the most important; then TLDs and on down, as each subzone decides it needs the additional security; deployment in a top-down fashion. Item (4) implies that in fact deployment will not be top-down, but rather bottom-up. We predict a model in which small organizations will decide to deploy DNSSEC along with IPSec, with VeriSign only moving to DNSSEC if and when enough bottom-up pressure has created a market.

VPNs have a proven market (corporate, military, and wireless) and so we can expect some adoption of a DNSSEC+IPSec package from smaller organizations. However, if VeriSign does not implement DNSSEC for the .com zone, it will be a significant deterrent to this process. In the next section, we speculate on a technique for speeding up a bottom-up deployment of DNSSEC.

6 Cross-organization deployment

The problem with a bottom-up deployment is that when two separate organizations have deployed DNSSEC, but their parent zone has not, each organization can use DNSSEC internally, but it is difficult to use across the organizations. Either each organization would have to configure their nameservers to accept each others keys, or they would have to convince their parent zone to move to DNSSEC.

We propose a new approach whereby two organizations can use each others nameservers securely, without manually configuring DNSSEC keys. An organization that would like to participate in our scheme would publish its DNSSEC key on its web site at a known location. For example, an organization example.com would post its key at https://www.example.com/dnssec.txt. The file could also contain additional DNS resource records from the example.com zone (such as hostnames and address of nameservers), if desired.

The resource records in this file would be used with a "trust, but verify" strategy. A party that would like to use the DNSSEC key contained in the file for making DNS queries for the example.com zone would retrieve the file using the HTTPS protocol, so that SSL would be used to protect the integrity of the file. In particular, the contents of the file would be signed, the signatures of the file and the certificate would be verified, and the retriever would make sure that the CA was trusted. The trust model (who is trusted) departs from that of DNSSEC, but is based on the familiar SSL trust model. We are thus using the existing SSL infrastructure to "bootstrap" the DNSSEC infrastructure across organizations.

Retrieving and validating the file could be integrated into the resolution process in a number of ways. For example, the file could be fetched on demand by the resolver. Another approach is to run a trusted search service that would do a web crawl for the key files of many organizations and cache them, and have the resolver query this trusted service as necessary.

Another variant would statically sign the files using the SSL key, and would place the file and certificate together on a non-SSL web server for a web crawler to find. This would have the advantage that the web crawler need not be trusted to do the validation, and the key could

not be tampered with on the web server without detection. Even an existing general search engine, such as Google, could be used for this.

As a final variant, we could place the signed key and SSL cert in DNS, using a new, distinguished resource record, and modify the resolution process to use the key if such a resource record is found and validates.

Economic analysis For such a service to succeed, one needs to find a sufficient number of early adopters such that Metcalfe's law can take a hold. As soon as there is a "critical mass" of users, the utility of the service will become sufficiently high such that virtually everybody needs to adopt the service. The adoption rate of an innovation can be characterized by five variables [9]:

Relative Advantage The advantage of our scheme over standard DNSSEC is that wide scale deployment is enhanced even without the cooperation of parties such as VeriSign. Combined with DNSSEC and IPSec, we have an advantage over SSL in providing VPN service at a lower level, hence, to more applications.

Compatibility The trust model of SSL is different from the DNSSEC trust model, however, it is well understood. The proposed scheme complements SSL and uses the same trust model as SSL. As an additional benefit, there is no need for a domain to purchase additional DNS related certificates.

Our service is also compatible with the existing DNS service. Applications can connect to a resolver augmented with our scheme just like it is a normal DNS service, and the service can use the DNS service for hostnames not yet published on the web.

Complexity Such a service would be easy to use because the user experience would not change. From an administrative point of view, the publishing of signed mappings and the populations of caches can be automated. The complexity of understanding the service is simplified by being based on well known concepts of web crawlers, caching, and SSL certificates.

Trialability In our scheme, domains can start publishing their signed mappings and use caches without the need for others to change their infrastructure. There is an immediate benefit within a domain and also between domains that decide to publish their mappings.

Observability Our scheme, combined with DNSSEC and IPSec, provides the directly observable benefit of VPN service within and across organizations. This is enhanced by well-publicized attacks or flaws in unprotected systems, such as wireless networks.

7 Conclusion

In an effort to produce a more secure Domain Name Service, DNSSEC has been designed and implemented over the last ten years. Even though DNSSEC adds value, we argue that it might not add sufficient value to justify the investment needed for a wide deployment. We suggest to bundle DNSSEC with IPSec to increase the value proposition of DNSSEC. Even though technologically quite different, we argued that SSL is a competitor of DNSSEC and not only of IPSec. We sketched a simple scheme for enhancing a bottom-up deployment of DNSSEC, which itself takes advantage of SSL and can be deployed without a wide scale infrastructure change. This combination might be even more economically viable than only combining DNSSEC and IPSec.

References

[1] Steven M. Bellovin. Using the Domain Name System for system break-ins. In *Proceedings of the fifth Usenix UNIX Security Symposium*, pages 199–208, Salt Lake City, UT, June 1995.

[2] Bind website. http://www.isc.org/products/BIND/.

[3] CAIDA. Nameserver DoS attack October 2002. http://www.caida.org/projects/dns-analysis/oct02dos.xml, 2003.

[4] CERT advisory CA–96.05—Java, 1996.

[5] Bill Cheswick and Steven M. Bellovin. A DNS filter and switch for packet-filtering gateways. In *Sixth USENIX Security Symposium*, July 1996.

[6] Drew Dean and Dan Wallach. Security flaws in the HotJava web browser. In *Proceedings of the IEEE Symposium on Security and Privacy*, May 1996.

[7] Warwick Ford, Phillip Hallam-Baker, Barbara Fox, Blair Dillaway, Brian LaMacchia, Jeremy Epstein, and Joe Lapp. Xml key management specification (xkms). http://www.w3.org/TR/xkms/, March 2001.

[8] James M. Galvin. Public key distribution with secure DNS. In *Sixth USENIX Security Symposium*, July 1996.

[9] Everett M. Rogers. *Diffusion of Innovations*. Free Press, 2003.

[10] Paul Vixie. DNS and Bind security issues. In *Proceedings of the Fifth Usenix Unix Security Symposium*, pages 209–216, June 1995.

[11] Paul Vixie, Gerry Sneeringer, and Mark Schleifer. Events of 21–Oct–2002, November 2002.

[12] Duane Wessels and Marina Fomenkov. Wow, that's a lot of packets. In *The 4th Annual Passive & Active Measurement Workshop*, 2003.

[13] Duane Wessels, Marina Fomenkov, Nevil Brownlee, and K.C. Claffy. Measurements and laboratory simulations of the upper DNS hierarchy. In *The 5th Annual Passive & Active Measurement Workshop*, 2004.