Emerging Standards
Editors: Tom Karygiannis, tomkary@nist.gov
Rick Kuhn, kuhn@nist.gov
Susan Landau, susan.landau@sun.com

Challenges in Securing the Domain Name System

Ramaswamy Chandramouli and Scott Rose
US National Institute of Standards and Technology

To an Internet-based application such as a Web browser or email, the Domain Name System (DNS) is no more than a look-up service for IP addresses. Given a user-friendly domain name such as www.nist.gov—also called a fully qualified domain name (FQDN)—the DNS provides the corresponding IP address (129.6.13.23 in this case). This process is called *name resolution*; a *DNS response* is the data provided in response to a name resolution query. DNS also performs other functions and provides data other than IP addresses, but it serves primarily as a name resolution service that handles *DNS query/response transactions*.

Unfortunately, the DNS has few security safeguards. In particular, there's currently no proof that the DNS server hasn't been corrupted. This has serious consequences for e-commerce and for the control of critical infrastructure. Imagine the economic impact if, for example, a rogue DNS server were to redirect Amazon.com customers to a fake Web site to which they submitted credit card and personal information to complete their purchases. In conjunction with various governments, research organizations, and the private sector, the Internet Engineering Task Force (IETF) has recently moved to address DNS security issues through the development of a specification and associated protocol called DNS Security extensions (DNSSEC).

DNS organization and infrastructure

Beneath the seemingly simple DNS look-up service lies a complex logical and administrative infrastructure. The DNS name resolution service uses a data repository, organized as a globally distributed database that's largely structured after the hierarchical domain name space, or DNS tree. At the top of the hierarchy is the root, a single domain represented by a dot (“.”). Below the root are top-level domains (TLDs), whether generic (for example, .com, .gov, or .edu) or country-specific (such ccTLDs include .de, .uk, and so on). The next level consists of enterprise-level domains (ELDs) owned by commercial, government, or academic organizations such as nist.gov or mit.edu. A large enterprise generally has administrative control over a given *zone*, classified as an ELD, and a set of subdomains (sometimes involving other related domains as well), such as csrc.nist.gov, cl.cam.ac.uk, or eecs.mit.edu. Thus, a zone refers to an administrative entity in the DNS that provides DNS services for a group of domains. The term “zone” has percolated up to the top levels, and the terms root zone, TLD zone, and ELD zones are often used.

As Figure 1 shows, the information flow in the DNS takes place primarily among three distinct system entities: authoritative name servers, stub resolvers, and caching name servers, also called resolving/recursive name servers or resolvers.

Every authoritative name server is associated with a zone and, as its name denotes, is the authoritative source for DNS data pertaining to that zone. To provide fault tolerance, effective administration, and efficient name resolution, several geographically and logically distributed authoritative name servers exist for each zone. These are further classified into primary name servers, which maintain authoritative DNS data in *zone files*, and secondary name servers, which frequently refresh their contents from the primary name servers.

Stub resolvers are lightweight clients that formulate DNS look-up queries on behalf of applications, such as Web browsers or email servers, and send them to caching name servers. Stub resolvers don't usually have any caching features.

Caching name servers each serve multiple stub resolvers. Depending on the zone or domain requested, they either query the appropriate authoritative name servers or serve responses from their own caches built from previous queries.

DNS security threats

Two main security threats exist for DNS in the context of query/response transactions. Attackers can

- spoof authoritative name servers responding to DNS queries and alter DNS responses in transit through man-in-the-middle attacks, and
- alter the DNS responses stored in caching name servers.

Hackers can exploit these threats to route Internet traffic away from its intended destination to malicious servers. Assuming that the usual host-level protection mechanisms exist to protect the stored data in the authoritative name servers and the cache in the caching name servers, the major security objectives for DNS clients are source authentication—ensuring the data received originated from an authoritative source—and data integrity—ensuring the data they receive hasn't been tampered with in transit.

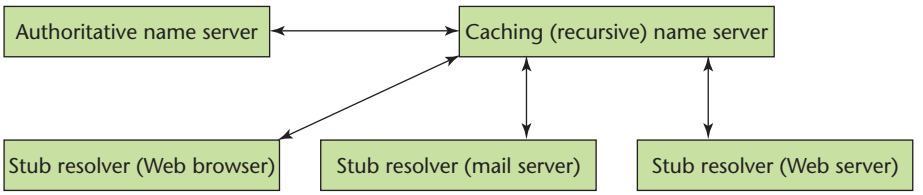

Figure 1. Primary Domain Name System (DNS) system entities. Authoritative name servers are the original sources for all DNS data for DNS administrative units, or zones. Caching name servers in an organization or ISP query authoritative name servers and cache data on behalf of stub resolvers, which are lightweight clients that formulate DNS look-up queries on behalf of applications.

Securing DNS

The IETF has defined the digital signature-based DNSSEC for protecting DNS query/response transactions through a series of requests for comments:

- RFC 4033 defines the security requirements for DNS, based on threat analysis;[1]
- RFC 4034 defines the necessary extensions to the existing zone file specification;[2] and
- RFC 4035 defines extensions to the existing DNS protocol to support the digital signature-based security specification.[3]

Although it's beyond the IETF's charter to mandate implementation of these specifications or provide the criteria for conformance, the US Department of Homeland Security has a proposal under way to include DNSSEC under the provisions of the Federal Information Security Management Act (FISMA), which mandates the implementation of security controls in US federal government information systems. Whereas FISMA applies only to US government systems, this move would enforce security for one of the largest subtrees in the DNS infrastructure, as well as serve as an impetus for private sector entities to deploy DNSSEC to do business with the US government.

Zones that implement DNSSEC are called *signed zones* because they include digital signatures for resource records in their zone files served by DNSSEC-aware authoritative name servers. In response to DNS queries, DNSSEC-aware authoritative name servers return *signed DNS responses* to DNSSEC-aware caching name servers. A signed DNS response contains three sets of records:

- the requested resource records (RRs),
- special resource records (SIG RRs) that carry the digital signatures associated with the requested RRs, and
- DNSKey RRs, which include the public key used to verify the signatures.

DNSSEC-aware caching name servers' ability to verify the signatures in signed DNS responses is somewhere between a full-fledged public-key infrastructure (PKI)-enabled server and a smart card. A smart card can authenticate an external entity by verifying an encrypted response only if the responding entity's public key is stored on the card itself. A PKI-enabled server can verify the digital signatures associated with messages from an unknown source, establishing trust in the source's public key through PKI-path validation by traversing a hierarchy of certificate authorities (CAs). A DNSSEC-aware caching name server, on the other hand, starts from a trusted public key stored within itself—the *trust anchor*—and establishes a trusted chain that ends in the public key of the zone that has provided the signed DNS response. A DNSSEC-aware caching name server can also use a *trust anchor list* to select different trust anchors for different DNS subtrees.

A DNSSEC-aware caching name server's ability to verify signatures in signed DNS responses from any given zone depends on its trust anchor list's contents. If the list consists of public keys high in the DNS tree, the caching server has a large zone population from which it can verify signatures. If the trust anchor list included the root zone's public key, the server could theoretically verify any signed responses because a path always exists from the root to any zone in the DNS tree. This path could become the trusted chain, provided that every zone in the path were a signed zone and carried the delegation information through the delegation signer RR (DS RR) for the next subzone in the path.

Deployment issues

Several issues remain before DNSSEC can be successfully deployed. The first is that the administration of large signed zones based on DNSSEC is challenging due to lack of agreement on best practices. Field data from deployed prototypes has simply

been insufficient, and it's been difficult to leverage knowledge gained from such infrastructures as PKI. The DNSSEC environment differs from a PKI-based environment in several ways:

- The signed zone doesn't use the concept of trusted third parties such as certificate authorities (CAs).
- The volume and frequency of signatures generated using a given cryptographic key—specifically, the private part of the public–private key pair—is larger than in traditional PKI.
- Using conservatively large key sizes—as in PKI—can negatively impact performance in zone signing and DNS response signature-verification processes.
- Certain secure practices adopted in PKI environments, such as keeping private keys offline, aren't possible for some authoritative name servers for which zone file contents must be dynamically updated online. Thus, private keys must be present online to regenerate signatures for modified zone file records.

DNSSEC needs to evolve its own policies and best practices for key size, key storage, and key life times for deciding on timelines for key rollovers. There are trade-offs between signature strength and DNS performance.

The second deployment issue is secure distribution and updating of trust anchors. DNSSEC-aware caching name servers depend on trust anchors to carry out DNS response signature verification, and DNSSEC deployment integrity depends on the presence of an infrastructure to securely distribute these trust anchors to the various DNSSEC-aware caching name servers. Networks switching over to DNSSEC-aware caching name servers for the first time can obtain these trust anchors as part of the software distribution. However, the public keys and their associated private keys that form the trust anchor list are likely to change periodically (key rollover) in the zones that own them. The main security challenge is to enable DNSSEC-aware caching name servers to securely update their trust anchors in response to key rollovers.

The major security objectives for DNS clients are source authentication—ensuring the data received originated from an authoritative source—and data integrity.

Researchers have proposed several solutions for the trust anchor update problem, but each has its drawbacks. We can use either a pull or push paradigm to distribute information between one-to-many or many-to-one entities. In a push paradigm, an authoritative name server that performs a key rollover must send the changed keys to all DNSSEC-aware caching name servers that use its public key in their trust anchor list. This is infeasible given that authoritative name servers don't maintain state information with respect to DNS query/response transactions, which can run into the millions and even the billions. On the other hand, employing a pull paradigm implies that every DNSSEC-aware caching name server runs an automated procedure for updating public keys such that the caching server must either poll relevant zones periodically or know the rollover schedules for their installed trust anchors. An inherent complexity in this approach is that zones perform both scheduled and emergency rollovers and often have two sets of signing keys—one for signing just the public key set and the other for signing the rest of the zone data.

Pilot projects in some European ccTLDs have demonstrated an approach to securely updating trust anchor lists by having a period of overlap during which clients can use both the old and new public keys with signed DNS responses. (See http://dnssec.nic.se/ for information on DNSSEC at the Network Information Centre, Sweden, and www.nlnetlabs.nl/dnssec/ for information on DNSSEC in the Netherlands.) Zone administrators use the existing private key to sign the new public key; the DNSSEC-aware caching name server can then use the new signature-verified public key to update its trust anchor list. One limitation is that this approach works only in situations in which a DNSSEC-aware caching name server is online when zones in its trust anchor list go through the overlap period for key rollovers—usually 30 days.

Another proposed solution is to create an out-of-band (outside of the DNS protocol) means, possibly a secure publish–subscribe protocol for distributing trust anchors that would let DNSSEC-aware authoritative name servers publish their keys to secure common locations from which DNSSEC-aware caching name servers could download them.

End-to-end secure DNS

Even after the community overcomes the preceding challenges and DNSSEC finds large-scale deployment, trust in DNSSEC ends at the caching name server because it's the entity that validates signatures in signed DNS responses. The next key step will be to expand DNSSEC, which is now limited to

the DNS infrastructure, into a specification for securing DNS from end to end. The caching name server forms the DNS infrastructure's boundary, but the specification needs to be extended to include the stub resolver, which shares the same address and execution space with the networked application.

An end-to-end secure DNS would let applications make decisions based on the nature of the DNS response. One way to achieve this is to incorporate the DNS response signature validation function into the stub resolver. The other option is to create a mechanism that lets the caching name server securely pass *DNS response security status*—information about the signature-validation process's outcome—to the stub resolver. The validation process could yield any of the following status options: unsigned response, validated signed response, failed signed response (sometimes called bogus response), or nonvalidatable signed response (that is, the caching name server doesn't have the right trust anchors in its list). In fact, the validating stub resolver could also generate this security status information. From the application's viewpoint, whether the caching name server or the stub resolver generates the information is immaterial. The application just needs the security status so it can decide, based on its own mission-critical nature, whether to use the DNS response. For example, a Web server might be willing to accept nonvalidatable signed responses, whereas a mail server might accept only validated signed responses. These end-to-end secure DNS proposals have come from the DNS research community.

End-to-end secure DNS standards

End-to-end secure DNS requires that stub resolvers perform signature validation or obtain DNS response security status from caching name servers. To realize either feature, the communication link between the two—the "DNS last hop"—must be secure. In situations in which stub resolvers and the caching name servers that serve them are behind corporate firewalls or communicate through virtual private networks (VPNs), the security of this link isn't an issue. Establishing a secure link isn't practically feasible, however, with caching name servers that serve public networks or stub resolvers in mobile devices that connect to different caching name servers each time. Even when a secure link is possible, there are currently no standardized formats or APIs to let caching name servers convey security status to stub resolvers or to let stub resolvers read and interpret them. In the latter case, the stub resolvers would have to perform the signature verification, which might require stub resolvers with larger footprints, adversely affecting performance, especially on small networked devices.

The US Department of Homeland Security is sponsoring an international effort to identify barriers and facilitate DNSSEC deployment to the global DNS tree, as well as to evaluate proposals seeking to address remaining technical challenges (www.dnssec-deployment.org). The DNSSEC-Deployment group's main focus is on promotion, education, and organizing research work on DNS security.

In addition to the issues relating to technical implementation and standards, there's also an economic dimension to securing DNS—especially end to end. Currently, no mechanisms or APIs exist that will enable networked applications to use signed DNS response validation outcomes. As a result, DNSSEC-aware networked applications aren't being developed. At the same time, there aren't any market drivers to extend DNS security mechanisms to the stub resolver because there's no perceived demand from the network application development community. ▢

Acknowledgments

Certain commercial entities, equipment, or materials are identified in this article in order to describe an experimental procedure or concept adequately. Such identification is not intended to imply recommendation or endorsement by the US National Institute of Standards and Technology, nor is it intended to imply that the entities, materials, or equipment are necessarily the best available for the purpose.

References

1. R. Arends et al., *DNS Security Introduction and Requirements*, IETF RFC 4033, Mar. 2005; www.ietf.org/rfc/rfc4034.txt.
2. R. Arends et al., *Resource Records for the DNS Security Extensions*, IETF RFC 4034, Mar. 2005; www.ietf.org/rfc/rfc4034.txt.
3. R. Arends et al., *Protocol Modifications for the DNS Security Extensions*, IETF RFC 4035, Mar. 2005; www.ietf.org/rfc/rfc4035.txt.

Ramaswamy (Mouli) Chandramouli *is a computer scientist with the US National Institute of Standards and Technology. His research interests include formal model-based security testing, smart cards, access control models, and security architectures. Mouli has a PhD in information technology from George Mason University. Contact him at mouli@nist.gov.*

Scott Rose *is a computer scientist with the US National Institute of Standards and Technology. His research interests include DNS security and measurement of discovery protocols. Rose has an MS in computer science from the University of Maryland, Baltimore County (UMBC). Contact him at scottr@nist.gov.*

Interested in writing for this department? Please contact editors Tom Karygiannis (tomkary@nist.gov), Rick Kuhn (kuhn@nist.gov), or Susan Landau, (susan.landau@sun.com).