Securing Web Presence with DNSSEC

By Peter Silva

This article discusses DNSSEC, a series of DNS protocol extensions which ensure the integrity of data returned by domain name lookups by incorporating a chain of trust into the DNS hierarchy.

Abstract

Domain Name Service (DNS) provides one of the most basic, but critical, functions on the Internet. However, security was not included in the original DNS design since scalability was the primary concern at the time of its creation. DNSSEC attempts to add security to DNS while maintaining backwards compatibility needed to scale with the Internet as a whole. DNSSEC is a series of DNS protocol extensions which ensure the integrity of data returned by domain name lookups by incorporating a chain of trust into the DNS hierarchy. Without DNSSEC it is possible for a malicious system to respond to a DNS query first with modified, false information. This article will discuss:

- The kinds of attacks that can exploit the DNS protocol such as DNS cache poisoning and how DNSSEC works to prevent these attacks
- The complexity, cost, and risks associated with deploying DNSSEC
- Infrastructure considerations that organizations looking to deploy DNSSEC must keep in mind
- How organizations can be prepared to comply with the upcoming U.S. government mandates for DNSSEC

Humans have always had difficulty remembering number sequences. Back in 1956, George Miller did some research on digit span recall tasks and found that humans are only able to hold seven plus or minus two items in memory. He concluded that even when more information is offered, the human memory system has the capacity to remember between five and nine chunks of information. Most people have a vocabulary of 10,000 to 30,000 words and some suggest that 500 to 1,000 of those are names. We have experienced words in place of numbers for years with the telephone system, especially 800 numbers, when a company spells out its product, name, or industry like 1-800-EAT-SOUP.

It should come as no surprise that when the Internet was invented with all its numbered Internet Protocol addresses, humans needed a way to translate those numbers into understandable names. The Domain Name System (DNS) was created in 1983 to enable humans to identify all the computers, services, and resources connected to the Internet by name. DNS translates human readable names into the unique binary information of devices so Internet users are able to find the machines they need. Think of it as the Internet’s phone book.

Now what would happen if someone changed your business name and matching phone book entry to his or her own? The phone book now lists “A. Crook,” an imposter who receives all of your calls and controls your number. Or, what if someone completely deleted your entry and no one could find you? That would really hurt business. What if that same situation happened to the domain name tied to your public website? An e-commerce site, at that! Either your customers will not be able to find you at all, or they will be redirected to another site that might look exactly like yours but is really A. Crook’s site. A. Crook happily takes their orders and money, leaving you with lost revenue, downtime, or any of the myriad other issues organizations face when their web property is hijacked.

DNSSEC

Security was not included in the original DNS design since at the time scalability – rather than malicious behavior – was
the primary concern. Many now feel that securing DNS would go a long way to securing the Internet at large. Domain Name System Security Extensions (DNSSEC) attempts to add security to DNS while maintaining the backward compatibility needed to scale with the Internet as a whole. In essence, DNSSEC adds a digital signature to ensure the authenticity of certain types of DNS transactions and, therefore, ensures the integrity of the information.

DNSSEC provides:

- Origin authentication of DNS data
- Data integrity
- Authenticated denial of existence

**Challenges**

**DNS in the wild: Bad things can happen**

DNS has worked just great since its inception, but as with almost everything Internet-related, the bad guys have found ways to exploit the protocol. One such way is called DNS cache poisoning. When you type a URL into your browser, a DNS resolver checks the Internet for the proper name/number translation and location. Typically, DNS will accept the first response or answer without question and send you to that site. It will also cache that information for a period of time until it expires, so upon the next request for that name/number, the site is immediately delivered. DNS will not need to query the Internet again and uses that address until that entry expires.

Since users assume they are getting the correct information, it can get ugly when a malicious system responds to the DNS query first with modified, false information as it does with DNS cache poisoning. The DNS servers send the user to the bad link but also cache the fake address until it expires. Not only does that single computer get sent to the wrong place, but if the malicious server is answering for a service provider, then thousands of users can get sent to a rogue system. This can last for hours to days, depending on how long the server stores the information; and all the other DNS servers that propagate the information can also be affected. The imminent dangers posed by a rogue site include delivering malware, committing fraud, and stealing personal or sensitive information.

In 2009, the main DNS registrar in Puerto Rico was hacked by a DNS attack.2 Local versions of the websites for Google, Microsoft, Coca-Cola, Yahoo, and others such as PayPal, Nike, and Dell were redirected to defaced sites or blank pages that told users that the requested site had been hacked. In this instance, the users were aware that they were not visiting the real site since the group who claimed responsibility gave notice. In a more sinister incident, one of Brazil’s largest banks suffered an attack that redirected users to a malicious site that attempted to install malware and steal passwords.3 In this situation, the users were not aware that they were on a fake site since the delivered page looked just like the original. These types of attacks are very hard to detect since the users had actually typed the correct domain name in their browsers.

**Taming the wild**

DNSSEC is a series of DNS protocol extensions, defined in Request for Comments (RFCs) 4033, 4034, and 4035, that ensure the integrity of data returned by domain name lookups by incorporating a chain of trust into the DNS hierarchy. The chain is built using public key infrastructure (PKI), with each link in the chain consisting of a public/private key pair. DNSSEC does not encrypt or provide confidentiality of the data, but it does authenticate that data.

DNSSEC provides the following:

- Origin authentication of DNS data: Resolvers can verify that data has originated from authoritative sources
- Data integrity: Resolvers can verify that responses are not modified in flight
- Authenticated denial of existence: When there is no data for a query, authoritative servers can provide a response that proves no data exists

Deploying DNSSEC involves signing zones with public/private key encryption and returning DNS responses with signatures (See Figure 1). A client’s trust in those signatures is based on a chain of trust established across administrative

---


Securing Web Presence with DNSSEC

A DNS zone is essentially a portion of the DNS namespace where administrative authority has been delegated, meaning some IT administrator is responsible for maintaining, updating, securing, etc., those specific DNS records. For instance, someone within F5 is responsible for f5.com but might have delegated authority to someone else for tech.f5.com or support.f5.com. F5.com is a zone itself and both tech and support could also be considered zones. The Root zone or top level domains (TLD) are the .com, .net, .org, .gov, and so forth. Second level domains are the names.com like f5.com. All the sub domains, like ask, tech, support, devcentral are all zones themselves. When resolving a name, you basically read right to left: check .com Root Zone to make sure f5 is a .com (and not a .org, .edu, etc.), then ask where to find f5.com, at which point Root will instruct where to find it – could be cached at a ISP or just tells the browser go find it at this IP address (which is the DNS server/database that keeps the record of where to find f5.com). Then, if necessary, it will ask where to find tech.f5.com and it gets resolved. Each zone is basically its own database with information on where to find the server housing the requested info.

DNS in practice (Figure 1)

In a normal DNS request, when a URL – example.com – is typed into a browser, the browser asks the local DNS server if it knows where the IP address for example.com resides. If it does know (the entry was cached), it will respond. If it does not, then it will ask the ISP DNS server which will either respond with the address or ask the Root DNS server (these are the Root zones). Since the requested name is a .com name, Root will tell ISP DNS to check with .com top level domain (TLD) DNS server. Top Level Domains are the .com, .net, .gov, .edu, etc. TLD .com server will tell the ISP DNS server to go ask “this” IP address, which happens to be the specific DNS server for example.com. That server will respond with the IP address (the specific server) to find the web page requested. During the process, someone could try to intercept the request and return an answer pointing to a rogue server. With DNSSEC, everything is essentially the same except that the DNS response must be signed. There is a chain of trust from the Root to the TLD to the specific DNS server for that name, and when it gets requested there is also a digital signature added to ensure that the response came from the authoritative server and not from some man-in-the-middle, providing assurance that the name came from where it is supposed to and that the browser is being directed to the authentic web page.

DNS zone

To accomplish DNSSEC:

1. Each DNSSEC zone creates one or more pairs of public/private key(s) with the public portion put in DNSSEC record type DNSKEY.

2. The zones sign all resource record sets (RRsets) and define the order in which multiple records of the same type are returned with private key(s).

3. Resolvers use DNSKEY(s) to verify RRsets; each RRset also has a signature attached to it that is called RRSIG.

If a resolver has a zone’s DNSKEY(s), it can verify that RRsets are intact by verifying their RRSIGs. The chain of trust is important to DNSSEC since an unbroken chain of trust needs to be established from the Root at the top through the top-level domain (TLD) and down to individual registrants. All zones need to be authenticated by “signing,” in that the publisher of a zone signs that zone prior to publication, and the parent of that zone publishes the keys of that zone. With many zones, it is likely that the signatures will expire before the DNS records are updated. Zone operators therefore require a means to automatically re-sign DNS records before these signatures expire. This functionality is called “continuous signing” or “automated key rollover” and is not yet a feature of common name server implementations.

Key Signing Keys or KSKs are used to sign other DNSKEY records and the DS records, while Zone Signing Keys or ZSKs are used to sign RRSIG. The KSK can be made stronger by using more bits in the key material. It has little operational impact since it is only used to sign a small fraction of the zone data and to verify the zone’s key set, not for other RRsets in the zone. The KSK should be rotated every 12 months and the ZSK every one to two months.

Since the KSK is only used to sign a key set, which is most probably updated less frequently than other data in the zone, it can be stored separately from and in a safer location than the ZSK. A KSK can have a longer key “effectivity” period. For almost any method of key management and zone signing, the KSK is used less frequently than the ZSK. Once a key set is signed with the KSK, all the keys in the key set can be used as ZSKs. If a ZSK is compromised, it can be simply dropped from the key set and the new key set is then re-signed with the KSK.

If a KSK is to be rolled over, there will be interactions with parties other than the zone administrator. These can include the registry of the parent zone or administrators of verifying resolvers that have the particular key configured as secure entry points. Hence, the key effectivity period of these keys can and should be made much longer. The public key enables a client to validate the integrity of something signed with the private key and the hashing enables the client to validate that the content was not tampered with. Since the private key of the public/private key pair could be used to impersonate a valid signer, it is critical to keep those keys secure.

©2010 Information Systems Security Association • www.issa.org • editor@issa.org • Permission for author use only.
Securing Web Presence with DNSSEC

Peter Silva

authentication of the signatures at the top level. Trust anchors add extra complexity and delays when deploying DNSSEC. When dot is signed, it will open the floodgates enabling smaller top level domains for countries around the world to easily sign their top level domain which in turn will enable sites in those domains to quickly and easily sign their own domains. It will have a domino effect on the DNS system and DNSSEC deployments.

Conclusion

DNS security and particularly DNSSEC has been cited by many publications as a Top 10 focus for IT departments in 2010. As more zones become signed this year, and as more companies recognize the importance (or are regulated to do so) of securing their DNS infrastructure, the DNS nightmare can hopefully become a DNS success story for 2010. DNSSEC ensures that the answer you receive when asking for name resolution comes from a trusted name server. It can help you comply with federal DNSSEC mandates and help protect your valuable domain name and web properties from rogue servers sending invalid responses.

Resources

—DNSSEC.net - http://www.dnssec.net/about.
—“You don’t have to wait to deploy DNSSEC” – http://blogs.techrepublic.com.com/security/?p=663.

About the Author

Peter Silva covers security for F5’s Technical Marketing Team. Starting out with a small VAR selling Netopia routers and the Instant Internet box, he soon became one of the first six Internet Specialists for AT&T managing customers on the original ATT WorldNet network. He now covers training, writing, speaking, along with overall product direction and evangelism for F5’s security line. He may be reached at p.silva@f5.com.

Solutions

DNSSEC, as originally conceived, was focused solely on traditional static DNS and never considered the requirements of Global Server Load Balancing (GSLB), or intelligent DNS. It is relatively easy to use BIND to provide DNSSEC for static DNS. It is more difficult to provide DNSSEC for dynamic DNS, and it is very difficult to provide DNSSEC for GSLB-type DNS responses, especially in cloud deployments. An ideal solution would be a device that signs DNS responses in real time and provides the means to deploy DNSSEC quickly and easily in an existing environment.

If a client requests a website that sits behind a DNSSEC device but does not request an authenticated answer, the DNSSEC device would do nothing and respond normally, passing through the DNSSEC device to the DNS server pool and returning directly to the client. When authentication is requested, the DNSSEC device should intercept the response and sign the response before sending it to the client computer. In this instance, the DNSSEC request passes through the DNSSEC device to the DNS servers. When the response is returned, DNSSEC signs the response in real time to ensure continuous signing. The potential attacker cannot forge this response without the corresponding private key.

This real-time signing is critical in dynamic content environments where both objects and users might be coming from various locations around the globe. There are devices that support DNSSEC for static DNS and DNSSEC compliance in general, but few have good solutions for dynamic content and none, so far, has any solutions for global server load balancing (GSLB)-type DNS responses in which the IP answer can change depending on the requesting client. Since GSLB can provide different answers to different clients for the same fully qualified domain name (FQDN), GSLB and DNSSEC are fundamentally at odds in the original design specifications.

A unique idea to the GSLB DNSSEC problem can address this by signing answers at the time the GSLB device decides what the answer should be, a real-time DNSSEC solution. Some feel a system in which every possible response is pre-signed might work but most have concluded that this is not a feasible approach.

The root zone signing schedule is targeted for July 2010.

Signing root aka dot is seen by the industry as a significant event. Current signed top level domains such as .org, .gov, .se, .us, have to rely on an external trust anchor system for authentication of the signatures at the top level. Trust anchors add extra complexity and delays when deploying DNSSEC. When dot is signed, it will open the floodgates enabling smaller top level domains for countries around the world to easily sign their top level domain which in turn will enable sites in those domains to quickly and easily sign their own domains. It will have a domino effect on the DNS system and DNSSEC deployments.