DNS Security: The truth is out there…but so are lies

David Piscitello, Core Competence, ICANN SSAC Fellow

The Domain Name System is arguably the most critical of all Internet applications. DNS is the network corollary to the automobile ignition system: if DNS isn't available, users are unable to resolve host names to IP addresses and thus cannot connect to Internet servers. Our dependency on domain name service makes the DNS a prime target for attackers. For example, many pharming and identity theft attacks poison name server records so that responses to DNS queries direct unsuspecting users to bogus web sites. Other attackers try to disrupt name service by launching distributed denial of service (DDoS) attacks against top level domain and root name servers.

Authentication, confidentiality, and integrity protection would make it difficult for attackers to exploit the DNS for pharming and identity theft purposes, but these security measures weren't among the original DNS design objectives and protocol. For some time, the Internet community has been developing - and deploying - measures to make DNS secure. Let's look at why these are important and how they can help mitigate several types of attacks.

"The truth is out there…but so are lies"

Dana Scully uttered this phrase in the X-files episode EBE, urging Fox Mulder to exercise caution during an investigation of UFO sightings that seemed to follow a semi (tractor-trailer) as it traversed the continental United States. Little did Scully know she was also commenting on the security of the DNS: the resource records that accurately associate a domain name with an Internet (IP) address *are* out there, but imposters who return false records may be there as well.

The threat vectors used to exploit the DNS fall into two categories: "liars" and "the lies the liars tell" (See RFC 3833). There are two kinds of liars. Both liars commonly masquerate or "spoof" the address of a legitimate host. For example, by impersonating a DNS client, an attacker can falsely register his computer and dynamically update misleading resource records at a DNS server. These liars are quite nasty, but nastier liars impersonate DNS servers so they can answer DNS queries and direct clients to phishing and identity theft web sites. Perhaps the nastiest of liars are those that impersonate DNS servers and transfer bogus zone information to an unsuspecting DNS server.

"The lies the liars tell" are the maliciously altered or *poisoned DNS resource records and zone files*. They include altered DNS cache data, subverted zone files, and DNS data intercepted and modified by an attacker during a man-in-the-middle (MITM) attack. Pharming attacks use several of these attack vectors.

To improve DNS security, we need stronger methods to detect and thwart impersonation than authentication based on IP addresses and we must be able to detect when DNS information has been altered without authorization, by a party other than the authoritative source.

"…we're alone on this. There's no one we can trust."

In EBE, Mulder and Scully argue whether to trust the mysterious old man, Deep Throat. Ultimately, Mulder and Scully agree to *trust no one*. However, Mulder and Scully did trust each other. The kind of trust they shared is the basis for performing authentication between DNS servers before either accepts DNS information. The DNS Security uses mutual authentication refers, a process by which two DNS servers trust each other only after each server proves to the other that it knows a secret that only the two servers share. This authentication method is described in RFC 2845, Secret Key Transaction Authentication for DNS. Here's how it works.

TSIG authenticates two name servers (A, B) that engage in a transaction – a DNS query-response exchange – as follows. DNS servers A and B are pre-configured with the same secret key. DNS server A computes a message digest or hash of a DNS message. (In the composition of this message, A includes a timestamp to protect against replay attacks as well.) DNS server A then digitally signs the message using the secret key it shares with B, and appends this digital signature to the DNS message in a special resource record, the Transaction Signature (TSIG RR) before sending the message to DNS server B. When DNS server B receives the message, B uses the secret key it shares with A to decrypt the signature in the TSIG RR to obtain the hash that DNS server A computed. DNS server B next computes the hash of the DNS message it received from DNS server A. B then compares the hash value it computed against the value DNS server A included in the TSIG RR. If the hash values are equal, then DNS server B knows that only DNS server A could have created the hash value attached to the DNS message and that the message B has received is an exact copy of that composed by A, i.e., that the message has not been altered in transit.

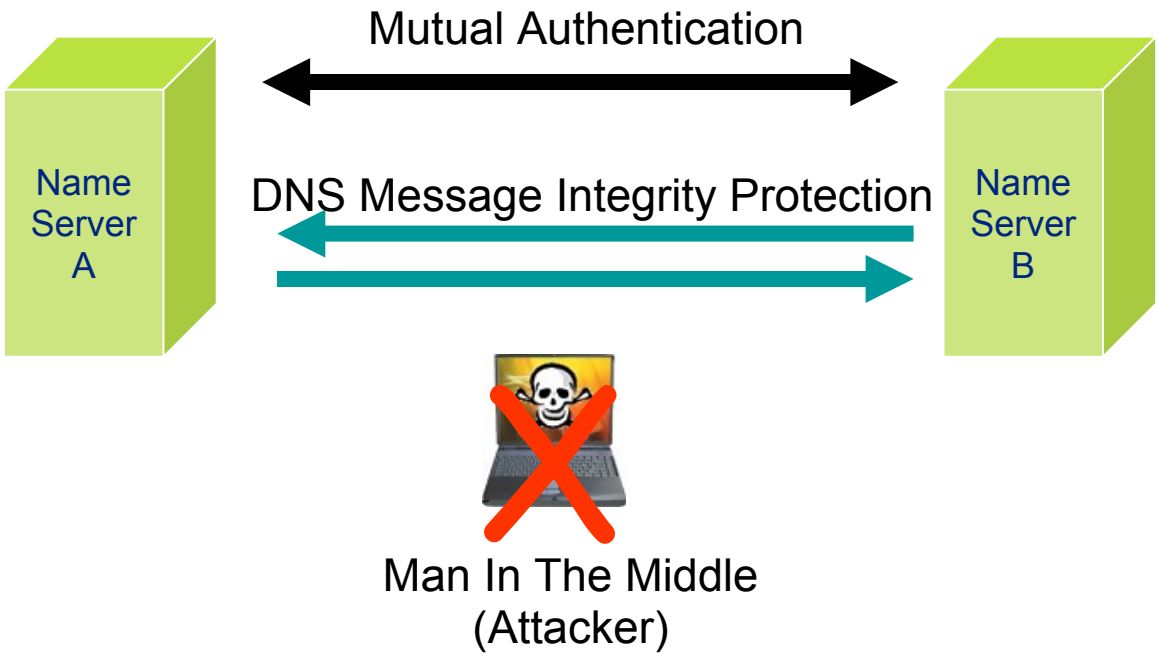

Figure 1: TSIG thwarts impersonation and message alteration by a man in the middle attacker

Note that while Figure 1 illustrates how TSIG authenticates and protects the integrity of DNS messages exchanged by two name servers, it could also be used to protect communications between a name server and a resolver (a software process or component of a user application that composes a DNS query message and processes the response message returned by a name server). TSIG can thus be used to protect a variety of DNS message transactions, ranging from a dynamic DNS update by a client host to a zone transfer.

TSIG uses shared secrets, so name server administrators must maintain a unique key for each name server "pair". Managing shared secrets does not scale well, so the most effective place to employ TSIG is at a "fan in" point in a network, where one or a handful of name servers process DNS queries for a very large number of resolvers. Thus, an organization might use TSIG to authenticate DNS transactions exchanged between resolvers and referral name servers it operates internally and one or more name servers operated by the organization's ISP. TSIG can also be used to authenticate dynamic DNS updates from approved DNS clients.

A final but important administrative matter to consider is that TSIG uses timestamps, so maintaining synchronized and accurate network time on DNS servers that use TSIG is extremely important. However, managing network time and coping with a modest number of shared secret keys is a small price to pay for the ability to thwart DNS message alteration and name server impersonation in strategic locations within your network.

… if I trust you, can I trust what you say to be true?

TSIG provides per transaction integrity protection and thus mitigates the threats of name server impersonation and DNS message alteration. In the DNS security world, this is sometimes called "channel" security. TSIG does not, however, provide the means to verify whether the DNS information conveyed in that transaction wasn't maliciously altered prior to transmission.

Like any other data, DNS data are vulnerable to corruption and unauthorized modification. For example, a successfully executed DNS cache poisoning attack will result in the corruption of DNS information that a name server records in its local cache. Similarly, a successfully executed "privilege escalation" attack can provide an attacker with full administrative control over an authoritative name server: acting as administrator, the attacker can modify individual name records or entire zones in that server's master file. The attacker could also impersonatea name server and return "non existent name" to any or all queries for this domain. Such attacks are sometimes called "Betrayal" attacks because a trusted source provides false DNS information.

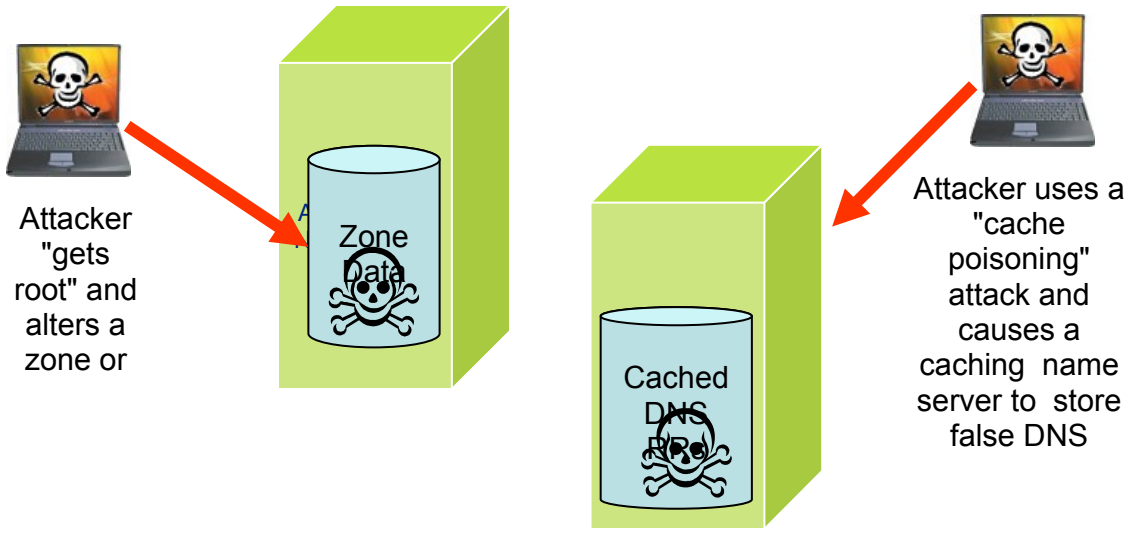

Figure 1. Examples of Attacks against DNS Data

To mitigate threats against DNS data "at rest", we need the authority that administers a domain's name data to provide *data object security*, i.e., data origin authentication and zone data integrity protection. Data origin authentication confirms that the name server claiming to be the authoritative source for a domain's zone data really is the authoritative server and not an imposter. Zone data integrity protection provides the recipient a guarantee that DNS data it receives are accurate and authentic copies of the domain's zone data.

DNS Security (DNSSEC, RFCs 4033, 4034, 4035) uses public key cryptography to hash and digitally sign a domain's zone data as follows. A DNS administrator creates a public/private key pair for zone data of a domain over which he claims authority. For example, the DNS administrator of `corecom.com` creates a "zone signing" key pair, stores `corecom.com's` private key in a secure manner and publishes the public key in a new resource record type (DNSKEY) in `corecom.com's` zone file. Publishing the public key provides a necessary and valuable scaling property. It allows any name server to acquire `corecom.com's` decryption key and thus decrypt any zone data that `corecom.com's` DNS administrator has encrypted using `corecom.com's` corresponding private key.

`Corecom.com's` DNS administrator next computes a message digest (also known as a hash) over `corecom.com's` zone data using a popular algorithm (typically MD5 or SHA1), and encrypts the message digest using `corecom.com's` private key. The DNS administrator adds this digital signature to `corecom.com's` zone file in another new resource record type, Signature (RRSIG). The RRSIG is then included in all DNS response messages.

A name server that receives zone data purported to be from `corecom.com` determines whether the data are authentic or altered by obtaining the public key from the `corecom.com's` DNSKEY record and then decrypting the hash value recorded in the RRSIG record using the algorithm indicated in the RRSIG record, producing a result we'll call "hash value received". The name server then computes a hash of the *zone data*

it received using the message digest algorithm indicated in the RRSIG record, producing a result we'll call "hash value computed". If "hash value received" is identical to "hash value computed" the name server concludes that the zone data it received are accurate and authentic copies of `corecom.com's` zone data.

A domain's zone signing key is a critical element of the DNSSEC's data object security, so there must be some means to determine whether the signing keys themselves are authentic. Each zone administrator arranges to have the domain parent sign his zone's public key and makes this signature available in a Delegation Signer resource record (DS RR). For example, Core Competence's domain label `corecom` is registered under `.com` so Core Competence have VeriSign to sign its public key with VeriSign's private key. Conceptually, VeriSign would have the public key it uses to sign `.com` signed by IANA, who administers the authoritative root ("dot" or "."). Name servers that implement DNSSEC can thus validate signatures by tracing the resulting chain of signed public keys back to a trusted root of security. In our `corecom.com` example and in the ideal deployment scenario, this is the authoritative root or ".".

DNSSEC's data object security measures complement rather than duplicate TSIG's channel security measures. Thus, an organization seeking to adopt a strong security policy should consider having its resolvers and name servers validate zone data whenever the RRSIG and DNSKEY records are availble, use the DS to validate signing keys, and use TSIG to insure integrity of communications with whatever set of name servers it chooses to trust.

Summary of DNS Security Measures

TSIG

- Uses shared secret keys to authenticates endpoints of a DNS transaction (e.g., query/response)
- Uses a message digest to provide integrity protection over DNS messages exchanged between authenticated endpoints

DNSSEC

- Uses private key to sign zone data
- Public key is published in DNSKEY RR so any name server can authenticate zone data origin
- A chain of delegation signers (DS) verifies that the signing key is authentic

Readers familiar with the "holy trinity" of cryptographic services may note that while authentication and integrity protection are provided by both TSIG and DNSSEC, neither provides a confidentiality service. This is deliberate. Future DNS Security Extensions will protect zone files during transfer. Organizations that do incorporate data in their zones that should not be publicly disclosed typically "split" DNS into public and private zones and carefully restrict transfer of the latter

A DNS Security Deployment Strategy

Many organizations are coming to terms with the fact that the threats of DNS-related attacks are sufficient to warrant attention. Here's one of many possible road maps an organization can follow to deploy DNS security measures:

1. Become familiar with TSIG and DNSSEC. Bookmark the web site DNSSEC.net and take advantage of the many useful resources there to learn more about the

protocols and the availability of TSIG- and DNSSEC-capable software, and to obtain valuable tools to manage your deployment.

2. Define a process for managing keys, distributing TSIG shared keys and publishing DNSSEC public keys for your organization. Evaluate the open source key management tools identified at DNSSEC.net or check with your DNS or server OS software vendor to learn what tools are available.
3. Obtain and install TSIG software. Distribute shared secret keys to resolvers and name servers that will use TSIG's channel security measures (one secret key for each pair of endpoints).
4. Obtain and install DNSSEC software. Create a keypair for your zone data and include the public key (DNSKEY) in your zone file. Sign and publish your signed zone data in your test environment.
5. Log extensively while you are experimenting with both TSIG and DNSSEC. Experiment until you are satisfied your installation and key management processes are stable before you install and deploy DNS security on your production name servers.
6. When you are ready to deploy DNS, publish your signed zone on your production servers and make your zone's public key available for others to validate your zone data.
7. Arrange for your domain's parent authority sign your keys.
8. Verify that name servers and security systems are capable of processing Extension mechanisms for DNS (EDNS0, RFC 2671). EDNS0 defines methods for a host to announce that it is capable of UDP-encapsulated DNS messages that exceed 512 octets, and the inclusion of DNSSEC resource records often results in large DNS response messages.
9. Verify that any security systems that inspect DNS traffic will allow messages containing the new resource records introduced in TSIG and DNSSEC. Some firewalls, DNS application proxies, and intrusion prevention systems may interpret these new resource records as hostile. In some cases, you will only need to adjust policy at security systems to allow these new records to pass. In other cases, you may have to check with your vendor to determine when DNS security will be supported.
10. Adjust logging levels for DNS security protocols so that it is consistent with your overall logging strategy.

Conclusion

We rightly and regularly complain that identity theft, pharming, and other attacks that exploit and misuse the DNS are huge problems, but to date, very few organizations have taken steps to mitigate DNS exploits within their own organization. Securing the DNS is a collective effort. Take responsibility for your part of the effort now.

Mulder and Scully would like DNSSEC. If DNSSEC were widely deployed, Mulder would never again have to wonder "which lie to believe".