Fault-Tolerant Mesh of Trust Applied to DNS Security

Wes Griffin Russ Mundy Sam Weiler
Network Assoicates Labs
wgriffin/mundy/weiler@tislabs.com

Dan Massey Naheed Vora
USC/ISI
masseyd/vora@isi.edu

Abstract

The Domain Name System is critical for the proper operation of applications on the Internet. Unfortunately, the DNS has a number of significant security weaknesses that can result in the compromise of web sites, email messages login sessions. Additionally, these weaknesses have been used as the basis for man-in-the-middle attacks on what are considered secure network protocols. This paper provides a short description of the weaknesses of the current DNS and a description of DNS security extensions that will solve the existing insecurities.
Keywords, Domain Name System, DNS, infrastructure, security.

1. Introduction

The current domain name system is insecure. When a user makes a legitimate DNS query to a name server, the DNS server will send a response back to that user. These queries and responses take the form of network packets that traverse a local network and, in many cases, will traverse the global Internet as well. A well positioned attacker can easily replace the legitimate responses from the server with whatever he chooses. In this attack, being well positioned means the attacker must be able to provide an 'answer' to the user before the legitimate DNS server. For instance, suppose a user in the umbc.edu domain requests information about the address for a search engine such as google.com or yahoo.com. Any user on the same local network segment as the user can readily provide a false answer to the unsuspecting user. Not only is this insecurity straight forward to exploit but there are tools such as dsniff readily available on the Internet (http://naughty.monkey.org/~dugsong/dsniff/)

In response to the insecurity problem, the DNS security extensions have been evolving in the Internet Engineering Task Force (IETF) community for number of years. These extensions add cryptographically based source authentication and data integrity. By adding source authentication and data integrity to the DNS, a user can cryptographically verify that a validated answer is the data that the domain administrator configured into the authoritative name server. That is, DNS security protects the data the zone administrator configured into the name server. Of course if the administrator configures the wrong data, the DNS security extensions cannot tell the administrator that the data is incorrect.

Another important feature of the DNS security extensions is that they do not modify the fundamental DNS protocol. DNS security specifies additional resource records for cryptographic keys and signatures and other supporting data that allows a secure DNS resolver implementation to verify the data received from the open Internet. The additional records do not restrict old resolver implementations from continuing use of the DNS protocol in an insecure manner. This design choice is a major benefit to DNS security as it allows security to be deployed incrementally and not require everyone on the internet to switch to a new DNS protocol all at once.

Without DNSSEC, the DNS provides no guarantees of data authenticity. An attacker can forge responses to DNS queries, allowing the attacker to easily redirect application packet such as HTTP requests, mail messages, and interactive login sessions. An attacker could then return forged data, capture passwords, eavesdrop on communications, and do more sophisticated man-in-the-middle attacks. Malicious attacks aren't the only threat: a misconfigured cache or replica (or secondary name server) can serve old, stale data, with no indication that it is out of date.

2. Approach

The DNS is divided into zones and each zone can be managed by different entities. For instance, com. is managed by VeriSign, which is responsible for assigning sub-zones to other entities such as tislabs.com. In turn, tislabs.com is managed by Network Associates Labs. Similarly, mil. is managed by the Defense Information Systems Agency and darpa.mil. is managed by the Defense Advanced Research Projects Agency.

This organization gives the DNS a tree structure where each node has a unique parent and may have any number of children. A zone's parent is usually indicated by the next label in the name. For example, the com. zone is the parent for tislabs.com. while mil. is the parent for darpa.mil. Similarly, darpa.mil. is also called a child of .mil. Even com. and mil. have a parent, it is called the root zone and is indicated by the right most '.' of a complete domain name. (It should be noted that this '.' is often omitted, i.e., disa.mil. is often written disa.mil with no right most '.'.) The root zone is special in that it represents the upper most part of the hierarchy and does not have a parent zone. DNS security specifies that security is provided at the zone level by the individual administrators in control of each domain name zone.

The DNS security extensions provide source authentication and data integrity through cryptographic signatures over the DNS data. The signatures are associated with sets of resource records. The signatures are generated by a private key that is held by the zone administrator. DNS security creates no constraints on how the private keys are managed. A number of suggestions have been put forth about storage and protection for private DNS keys but the final decision about amount and type of private key protection rests with each zone administrator. The public key, however, must be published in the DNS and served by a DNS name server. The public key is signed as well. This method of signing the data and then publishing the signatures as well as the corresponding public key allows a secure resolver implementation to obtain the public key from the DNS, verify they signatures on the data that it receives in responses from the name server, and then cryptographically know whether the data is correct or has been modified in some way.

Key distribution, for the most part, will be handled in-line using the normal DNS protocol. Public keys are published along with other DNS data and resolvers can request those keys and subsequently verify the signatures. To achieve this security a resolver must obtain the public key for a zone in a trustworthy manner. This might not be a problem for two organizations that are working closely since the public keys can be exchanged in a trusted manner and then configured into the DNS security aware resolvers. If the user of the DNS does not already have a relationship with the owner of the other zone, the solution is more complex. A parent will use a resource record to indicate which child zone public key should be used to verify the child's data. If the user has securely obtained a parent zone public key, then the user can verify any child zone data. In the easiest case, the resolver has the public key for the root zone and is able to obtain and validate the information for each zone until it reaches the zone requested by the user. The specific methods for obtaining the public key for the root zone are currently being discussed in the IETF community.

As an example, here's what happens when a DNSSEC-aware resolver queries for www.tislabs.com. with the assumption that the root is signed and the client has the root's public key. The resolver first goes to a root name server, which returns a list of name servers for .com. as well as hash of the key used to sign .com. and a signature for that key. The resolver can verify that signature using the well-known root key that is statically configured in it. The next queries are to the name servers for .com., which return a list of name servers for tislabs.com as well as a hash of the key used to sign tislabs.com. and a signature, name by .com.'s key, of that hash. The client can also retrieve .com.'s public key, comparing it to the signed hash obtained from the root name servers, and then using it to validate the signature of the hash of tislabs.com.'s key. The process repeats once again, this time with the query directed to tislabs.com.'s name servers. This time a signed answer (either a positive answer with an A record or a proof of non-existence of such a record) is returned. Again, the client can fetch tislabs.com.'s key, compare it to the signed hash obtained from .com., and then use it to verify the final answer.

The original DNS protocol makes heavy use of caches for a number of reasons. Most importantly, they reduces the load on the authoritative name servers. Caches also allow an authoritative name server to be "down" for whatever reason, and the name can still be used by resolvers that use caches. Another major benefit of the DNS security extensions design is that it does not affect the caching properties of the original DNS protocol. Caches will still function without any modification with the DNS security extensions enabled on a zone. This ties in with the decision to not modify the original protocol. It allows individual resolvers and name servers to be upgraded piecemeal to DNS security-aware implementations without affecting real-world operations.

3. Summary

Without DNS security, the DNS provides no guarantees of source authenticity or data integrity and misconfigured name servers can server old, stale and out of date data with no indication that that the data is out of date. To make matters worse, an attacker can forge responses to DNS queries, allowing the attacker to easily redirect application packet such as HTTP requests, mail messages, and interactive login sessions. An attacker could then return forged data, capture passwords, eavesdrop on communications, and do more sophisticated man-in-the-middle attacks. DNS security provides the source authenticity and data integrity needed to solve these shortcomings.

4. Demonstration

In this demonstration, we will demonstrate both the vulnerabilities in the current DNS and the counter-measures offered by DNS Security. In our demonstration, we use DNS attacks to redirect traffic to our "attacker". When then apply the DNS security extensions and use tracking software developed under the FMESHD project to show how the attack is both detected and the correct DNS data is discovered.

But as with any security extension, the overall added security is only effective if the system can be operated correct Major advances in DNS security address the operational concerns and we demonstrate techniques for securing DNS zones and operating secure zones. Using toolsets developed under the FMESHD project, we show how an administrator can easily transition to a secure DNS and then maintain secure DNS zone in a relatively simple and automated fashion.