Minimizing Information Leakage in the DNS

Scott Rose and Anastase Nakassis, National Institute for Standards and Technology

Abstract

The domain name system is the global lookup service for network resources. To protect DNS information, the DNS security extensions have been developed and deployed on branches of the DNS to provide authentication and integrity protection using digital signatures. However, signed DNS nodes were found to have an unfortunate side effect: an attacker can query them as reconnaissance before attacking hosts on a particular network. There are different ways a zone administrator can minimize information leakage and still take advantage of DNSSEC for integrity and source authentication. This article describes the risk and examines the protocol and operational options and looks at their advantages and drawbacks.

The Problem

Recent additions to the DNS protocol to add origin authentication and integrity to DNS data (referred to as the DNS security extensions [2–4] or DNSSEC for short) has had an unfortunate side effect related to zone enumeration. DNSSEC adds three new resource record types (RR types) to store public keys, hashes, and digital signatures, as well as a fourth RR type to handle authenticated denial of existence. This RR, called the next secure RR (NSEC RR) is used to provide digitally signed responses for error situations, such as a client query for a non-existent name.

The NSEC RR provides proof of non-existence by providing two valid, ordered names in the zone between which there are no valid domain names. For example, if a client queries for the IP address of the host b.example.com and gets back an NSEC RR with a.example.com and c.example.com, the client can deduce that the name b.example.com does not exist as it falls in the span bookended by a.example.com and c.example.com. As with all DNSSEC responses, a separate digital signature accompanies the response to prove the data came from an authoritative server for the zone.

As a side effect, the client now knows two names that do exist in the zone that it might not have known before: a.example.com and c.example.com. The NSEC RRs in a zone form a chain to cover the entire name space — linking every name through NSEC RRs. Using this information, an attacker can start to “walk” the zone and send a subsequent query to find the name that exists after c.example.com all the way until the NSEC chain loops back to the top of the zone. For a zone with N names, it would take N queries to obtain a list of every name in the zone and any other information stored in the DNS. An attacker can then use this list in planning an attack against individual hosts.

One early response to this was to have “minimal spanning NSEC RRs,” described in [5]. The idea is to have a server generate a set of NSEC RRs based around the error producing query name. This would increase the time in enumerating the zone using NSEC RRs to the same as a direct brute force attack against the server as the span covers only the error-generating query name; the attacker does not gain knowledge of hosts that do exist in the zone. This variant requires a serv-
er to generate and sign a unique response for every error message, which could quickly become a denial of service (DoS) attack against the server that may do more harm to a network than a zone enumeration attack.

That does not mean there must be a trade-off between the content protection DNSSEC provides and reducing DNS information exposure. There are other methods zone administrators can utilize to minimize zone information leakage and still have the benefits of DNSSEC signed zones. One such method is using a variant of the NSEC RR known as the NSEC3 RR. The other is using operational architectures to insure that only those hosts that must be publicly accessible can be found in the global DNS, whereas private hosts are kept in a separate name space isolated from all but those clients that must query them.

**NSEC3 Variant in DNSSEC**

The NSEC3 resource record [6] (NSEC3) is a recent variation of the original NSEC RR in DNSSEC. It was developed to add some obfuscation to the domain names in the NSEC RR to make zone enumeration a more difficult task. Its format is identical to the NSEC RR but with hashed domain names (using a one-way hash function such as SHA-1; see Fig. 1). That way, a client can still determine that the query name does not exist (the hash of the query name falls in the span between the two hashed names provided), but not learn about any valid domain names that do exist in the zone. The NSEC3 RR is used exactly in the same way as the NSEC RR but requires both the server and the resolver to be able to perform the hash function used (by default — SHA-1, but SHA-256 and other hash functions can be defined).

There are additional fields in the NSEC3 RR to declare the number of iterations and salt value to be used with the hash calculation. The salt value is a hex string that is to be appended to the query name before hashing, and the iterations field is used to indicate how many times the hash function is to be computed for a give name. Both values can be changed periodically to compensate for increased computing power available to an attacker.

The NSEC3 RR is not a perfect solution, because existing domain names are still being leaked with every name error response. Depending on the error response, multiple NSEC3 RRs may be returned, each with two (hashed) real names that exist in the domain. This means that it is still possible to get a complete list of hashed names in the zone. Unlike DNSSEC with NSEC, an attacker cannot always choose which queries to send to obtain all of the names, but we can calculate how many queries on average. While DNSSEC with NSEC requires $N$ queries to map the names of a zone with $N$ names, in the naive case where hashed names are evenly dispersed throughout the (hashed) name space, DNSSEC with NSEC3 requires an attacker send $O(N \ln(N))$ queries to obtain the same list of $N$ names and in cases where the hashes are not evenly distributed, the required number of queries can reach $O(N^2 \ln(N))$, which could still be manageable for a small number of names [7].

In practice, the number of queries and the time required to conduct this attack depends on the computing power of the attacker, as well as the network delay and the number of unique domain names in the zone. An attacker can reduce the time and number of queries required by pre-computing the hash of possible query names and rejecting those that fall in known NSEC3 spans. This only applies if the work required to construct the hash is less than the time required to conduct a DNS transaction. That is, the zone administrator chooses iteration values low enough that it is cheaper to generate a hash than to send a query for a random name to the DNS server.

If an attacker can conduct these queries in parallel (using a botnet for example), it is still possible to obtain a list of hashed names for a particular zone in a relatively modest amount of time. The attacker can then perform a brute force attack with the list stored locally (and avoid unnecessary network traffic that may trigger a security monitoring alert).

This is where the NSEC3 iterations and salt field come into play. The salt value is a random string chosen by the zone owner. The purpose is to prevent an attacker from pre-computing a dictionary of all possible names to compare against hashed names discovered in NSEC3 RRs. If attackers cannot predict the salt value used, they cannot construct a dictionary and may be required to reconstruct the list of hashed names when the salt changes — unless they are sure they have the entire list of names. The iterations value is used to slow an attacker when conducting a brute force attack against a list of discovered hashed names. Increasing the iterations value increases the number of times the hash should be computed. The goal is to make an offline attack as time consuming as conducting a brute force dictionary attack against the server. However, choosing an iteration value that is too large (making it more expensive than a brute force attack against a server) also slows the server itself (and all clients) who must also perform hash calculations to form (and validate) responses using NSEC3 RRs.

An attacker does have several options available that can defeat high iteration values in NSEC3 RRs. First, attackers may have more computing power available than the zone owner has, either through more powerful hardware or through a network of computers (such as a botnet) that enables them to attack the list of hashes in a distributed manner. Second, the attacker need not necessarily conduct a brute force attack. The nature of DNS is that it is used to map human readable names to IP addresses or other network resources. Therefore, there is more likely a subset of all possible name combinations in use for names in the zone. For example, the majority of zones has at least one host called “www” and may have another host named “mail” or “router,” and so on. Previous network studies have compiled lists of the most common names found in zones that an attacker can use to quickly discover if certain common names are among the list of hashed names obtained from the zone [8].

**Split DNS**

Another option to prevent zone content from attackers wishing to map the resources of a network is simply to remove those names from the global DNS. Most large organizations already separate their internal network from the global Inter-
Recent security extensions to the DNS (DNSSEC) have been developed to provide integrity and source authentication for DNS zone data. However, this attempt to prevent one class of attacks (client redirection) has inadvertently made a new type of attack on the DNS possible — zone enumeration. This attack was always theoretically possible but was too expensive to perform compared to the gain. Whereas enumeration may be a direct risk to only a few zones, it is often the start of a directly targeted attack. There are multiple ways to minimize this risk; each with advantages and drawbacks.

The most direct solution is to have the DNS server generate a NSEC RR for each negative response that shrinks the span so that only the non-existent query name is covered. This would force an attacker to expend the same number of queries as a direct brute force attack. It also would force a server to generate NSEC RRs (and sign them) during run time, which could quickly become a DoS attack if the attacker could send a large volume of queries.

The NSEC3 variant of DNSSEC was developed to minimize zone information leakage and zone enumeration attacks. NSEC3 RRs use hashed domain names instead of cleartext domain names when forming negative answers in the DNS. However, it does not make it impossible to discover the names in a particular zone database. It only requires more work by the attacker to learn about actual host names by first obtaining the list of hashed names and launching an attack against that list.

The operational means to minimize zone enumeration is to keep separate zone files for internal and external hosts. Most organizations do this to some extent already by keeping systems they wish to remain private behind a firewall.

This can be done as a separate view (to keep the same domain suffix) or as a separate zone or subzone. Because outside attackers have no knowledge of the existence of the internal zone and cannot access the internal network, it becomes impossible to enumerate from outside the firewall. External servers are meant to be public, and it should not matter if the external network is enumerated. However, this set up requires much more planning and effort on the zone administrator’s part and makes DNSSEC deployment more difficult to maintain if both internal and external zones are to be signed. Have...
ing DNSSEC also makes an insider attack possible if an attacker can replay external zone error messages inside the private network. This would result in a DoS attack against the internal network client.

The type of defense to select depends on the organization. Because most organizations already have a split network, extra care to insure that only public servers appear in the external DNS makes sense. For zone administrators that have privacy concerns above zone enumeration, NSEC3 deployment may be the only choice. The DNS was designed to be public, and there is no way (short of using encrypted, secure network transmission) to totally prevent a zone enumeration attack. The only possible goal is to minimize its effectiveness.

Acknowledgments

This work was funded by the Department of Homeland Security Science and Technology Directorate.

References


Biographies

SCOTT ROSE (scottr@nist.gov) received a B.A. in computer science from the College of Wooster and an M.S. in computer science from the University of Maryland, Baltimore. He works as a computer scientist at the National Institute for Standards and Technology (NIST). He was on the editor team that produced the latest version of the DNS security extensions (DNSSEC).

ANATASE (TASSOS) NAKASSIS (tassos@nist.gov) studied mathematics at the University of Athens and the University of Paris (VI), obtaining a doctoral degree in algebra, and at the University of South Florida, obtaining a doctoral degree in probability. He is employed by NIST and is currently working on problems that straddle mathematics, information theory, and algorithm development with an emphasis on the classical aspects of quantum cryptography.