DNS Spoofing and Poisoning: Trust, Danger, and Solutions

Alex Polyakov

alexander.polyakov@tufts.edu

Mentor: Ming Chow, Department of Computer Science, Tufts University

Abstract. The Domain Name System is the way in which human-readable domain names resolve to numeric IP addresses. This protocol is meant to facilitate an easy, streamlined navigation pattern for human users across the web. Unfortunately, the user-friendly choices made in the design of the protocol come with tradeoffs when considering the security qualities thereof. This paper is meant as an overview of the threats facing the protocol, and by extension users on networks where DNS is in use as an address resolution mechanism. We will explore methods of attacking the system as well as potential motivations and benefits for attackers. Finally, the paper will propose several techniques defend the system against attacks, both on the level of individual users or providers, and on the level of protocol features that govern the functionality of DNS.

1. Introduction – The Domain Name System

The Domain Name System (DNS, for short), is the application-layer protocol which governs the way that Human-Readable addresses (i.e. a URI in the form "http://www.google.com") resolve to numeric IP addresses for the purpose of sending requests and responses along a network. The chief use of DNS is to provide human users of the Internet with a way to access the network, but abstract away the necessity for users to memorize the actual numeric identifiers for the resources therein. Often, a network DNS is likened to a physical phonebook, where human-readable names are also translated to numeric identifiers for navigation of the network (i.e. "John Smith – 617-111-1111"). There are, two significant differences between DNS servers and physical phonebooks: the concept of authority, and the ease of update. Phonebook authority is expensive to fake, because of the real costs involved in producing a large number of phonebooks for an attack, and the physical distribution thereof; faking authority of a DNS resolution requires a single point of fraud, and propagates to all requesters to the server. Essentially, DNS and phonebooks are differing relations – DNS is a one-to-many relation between resolver and requester, and a phonebook is a many-to-many relation. Furthermore, phonebooks, as physical entities do not often update, and are used infrequently, whereas DNS servers constantly update with new information from the governing servers on the network, and those changes are seen instantly, as a huge amount of DNS resolution

requests can exist on a sufficiently large network at once. Thus, even a brief incorrect entry in a DNS table can affect many users at once.

1.1 DNS Resolution

Simply put, the responsibility of DNS resolution is distributed across the network, under the purview of authoritative name servers. This distribution ensures resiliency and performance – if one server fails, there are others to pick up the load; if one server is too far away on the network, or has too high a request load to handle, other servers exist to provide answers to the request. DNS servers interact with each other according to a tree model, and to minimize traffic on the network and make sure that responses to DNS resolution requests are handled as fast as possible, each node in the recursive DNS resolution chain keeps a copy of the recent previous resolutions. A typical recursive resolution is processed as such:

- A client (i.e. web browser) initiates a request to the local DNS resolver (typically an ISP) to know the IP address associated with some URL
- The local resolver looks it up in its cache, and if the requested URL is there, the local resolver returns with the IP address associated with the URL. Otherwise, it sends a recursive request to the local DNS server.
- The local DNS server checks its own cache, and if the entry is present, it responds to the local resolver, which forwards the answer to the client that initiated the request. If the entry is not present in the local DNS server's cache, the local DNS server will issue a request to the root servers it communicates with to determine which server does hold the address associated with the name (authoritative server)
- Once it gets a response from the authoritative server, the local DNS server will cache that association (to ease future lookups), and forward that information back to the local resolver.

- The local resolver with also cache that information, and forward it to the client that had initiated the request.

1.2 Spoofing and Poisoning

Because DNS (as well as other protocols governing the function of the Internet) was designed for a network that was not open to the general public, security concerns were not major considerations for the protocol. As such, the design of the system leaves fairly significant vulnerability issues. For example, DNS cache poisoning – the distribution of data to caching resolvers that is not accurate. Because of the design of the DNS system, if this information is not verified in some way, and it often is not – the DNS specification makes no requirement for authentication of a response, then requesters will get information that they believe to be valid, but is in fact not. Since any resolver (local, ISP, or authoritative) in the recursive chain for the DNS resolution request can answer the request with the information in its cache, and the request will stop, thinking it has found the appropriate resolution, there exist several attack vectors by which a malicious agent can "spoof" a DNS resolution request. If any resolver at any level has incorrect information, and no step in the recursive DNS request has found an answer before that level of the recursion, then the (wrong) answer from that server will be the one that is used by the initial requester, and will be cached by the resolvers between the initiator and the nodes up to the poisoned node – meaning that not only will the first request resolve to a fallacious address, but also that any requests to resolve the same domain name the are initiated after the initial but before the caches of the intermediate nodes are purged of the false record will *also* resolve to the false record. Truly, poisoning a cache, especially at any level of height is a potent attack.

2. To The Community

Why choose to discuss DNS cache poisoning? As mentioned before, the system was designed with little consideration given to the security implications and potential vulnerabilities therein. While it's understandable that the system that DNS was designed for was not widely accepted by any means, and that more important than security was functionality and potential extensibility, now that the system of the Internet has evolved past the initial scope of its design, we must reevaluate the aspects of the system that have come under its new, broader scope. There is much to be gained by exploiting weaknesses in any distributed system – especially one as rapidly-propagating as DNS. Further, since DNS is based on trust between requester and resolver, and there is no information authentication built into the protocol, once access is gained to a given DNS caching resource, much damage can be done without even the realization that the system is being attacked.

2.1 Motivations and Methods for Attackers

Why go through the trouble of poisoning a DNS cache? One explanation is the enormous potential for phishing attacks once one can deceive users as to which host they're sending information on a server to. For example, if one gains access to a DNS resolution server's cache and changes the entry for "http://www.bankofamerica.com" to resolve to a numeric IP pointing to an attacker's webserver, and that attacker has configured the server to serve pages that look exactly like the homepage for Bank of America, then victims of the DNS poisoning attack (those who've requested for DNS resolution during the window when the DNS server's entry for that URL contains phony information) will see and be likely to enter their credentials on a phony phishing webpage. Using this technique, attackers can use DNS poisoning to facilitate social engineering

schemes such as phishing to obtain sensitive information from users that they can use for nefarious purposes.

Another possible exploit is the proliferation of malware. Using the same technique as above to direct potential victims to their server, a DNS poisoning attacker can host malware on a server pointed to by an inaccurate DNS name resolver entry. Unsuspecting internet users will be directed to the malware hosted on the attacker's server and be exposed to it. Again, as mentioned above, if DNS poisoning occurs at a node sufficiently high on the recursive DNS resolution structure, a significant number of users will be directed and exposed to the malware (and the DNS servers that refer to the poisoned one when they don't have the answer to the resolution of that URL will also cache the poisoned entry – recursively down to individual clients, spreading the damage).

Another method of DNS spoofing involves an attacker sending DNS resolution packets (with spoofed IP sender addresses, to make it seem like the packet came from another DNS resolution server) him- or herself, meant to arrive before the DNS resolution packet from the legitimate server. This way, any subsequent (correct) packets that come to the requester will be discarded, since the requester will have already gotten an answer, and the attacker will still have accomplished the goal of providing incorrect information to victims. From here, the attacker can accomplish the same ends as directly altering the cache of a DNS server, only without having to have gotten access to the DNS server's cache. Furthermore, this attack becomes DNS poisoning as well, if the requester is itself a DNS server, as the false packet will cause the requester to hold on to the false information, thereby poisoning its own cache.

3. Defending Against DNS Poisoning Attacks

The most readily-available defense against DNS poisoning involves securing the attack points on network infrastructures themselves – the DNS name servers. Proper use of security tools such as

firewalls, and patches of known vulnerabilities should be applied in a timely manner. Basic security savvy can go a long way to securing even something as critical as the DNS infrastructure of a network.

A particular technique to foil DNS spoofing, but not cache poisoning, involves randomizing the source port on the DNS requester – this way, a DNS packet that does not come from a trusted source (i.e. an attacker trying to send a DNS resolution packet to a requester before the actual answer gets there) will have a $1/2^{16}$ chance of going to the right port and the requester will know that any DNS resolution packet going to an incorrect port is an attack and must be discarded.

Alternatively, DNS resolution software can be implemented to poll multiple other DNS servers in the event that the resolver running the software does not have information on a particular DNS resolution. This way, any DNS server returning an entry that does not match what the majority of DNS servers respond with can be known to be malicious and its responses ignored (and if it is a known server, steps can be taken to secure it). In order to foil such a security measure, an attacker would have to control more than half of the servers in a given DNS "zone".

4. Summary

Any system built upon trust has the benefit of simple implementation, but at the expense of potential exploitation. This is very useful when growing systems, and all users of that system can be accounted for. However, the Internet has grown far, far beyond a small system whose users are all known, and as such systems that are built upon trust between the users no longer fill its needs. Specifically, DNS, a protocol which makes authentication optional and propagates information rapidly and broadly is a trifecta of security concerns. Attackers stand to gain a lot by exploiting the trust that Internet users put into the DNS system. Spoofed packets and altered cache entries are

just two of the ways that attackers can exploit DNS and trick users into either giving information to unsavory webpages or downloading software that can compromise their system. It is imperative that we re-examine our use of DNS and explore possibilities to defend ourselves against such attackers.

References

Couch, Alva. *Names and* Numbers. Lecture slides from COMP112 at Tufts University, Spring 2013. <http://www.cs.tufts.edu/comp/112/notes/Names_and_Numbers.pdf> Accessed Dec. 10, 2013

DNS Forgery. DNSCurve: Usable Security for DNS. <http://dnscurve.org/forgery.html> Accessed Dec. 10, 2013

Steinhoff, Wiesmaier, Araújo. "*The State of the Art in DNS Spoofing*". Department of Cryptography and Computeralgebra, Technische Universität Darmstadt. 2006

Postel, "RFC 1591, *Domain Name System Structure and Delegation* (Informational)", Network Working Group, 1994

have2Banonymous. "*The Impact of RFC guidelines on DNS Spoofing Attacks*". Phrack magazine. <http://www.phrack.org/issues.html?issue=62&id=3&mode=txt> Accessed Dec 11. 2013