Root Services Blog, Vol. I, No. 1, 2014
Copyright Root Services, 2014

Man In the Middle (MITM) DNS Spoofing Explained

Alexander Chopra[1]*, Michael Kaufman[2]

Abstract
This paper provides a high level understanding of utilization of DNS spoofing as a MITM technique. The paper explains how false DNS information is provided to a host so they are re-routed to a false source. The process as well as possible defenses are outlined in this paper.

Keywords
Man in the Middle — MITM — DNS — DNS Spoofing

[1] *Research and Development, Root Services, Rockville, Maryland, United States*
[2] *Research and Development, Root Services, Rockville, Maryland, United States*

Contents

Introduction 1
1 Domain Name System (DNS) 1
1.1 Utilization 1
1.2 DNS Software 2
2 DNS Spoofing 2
2.1 DNS Spoofing Process 2
2.2 Birthday Attack 3
3 BIND 3
3.1 Post-Attack 4
4 Other DNS Spoofing Applications 4
5 Keeping up to Date 4
6 Conclusion 4
Acknowledgments 4
References 4

Introduction

The Domain Name System (DNS) is a naming system for devices and services that are connected to an internal or external network. The DNS system can easily equate to a phone book which provides translation services from hostnames to IP addresses. To resolve the domain name google.com, DNS is used to map google.com into an understandable IP (Internet Protocol) address for your device, 74.125.228.69. The device is able to connect to the IP address at which google.com is located as a result of the DNS service correctly providing the information to where this hostname is pointing. DNS is used almost everywhere because of the ease of use that it provides to the average user. A user would not want to remember a bunch of IP addresses to access the information desired. An easy to remember domain name is much more convenient for a user to remember in comparison with a string of numbers. DNS names take the format of a name followed by a period and followed by a TLD (Top-Level Domain). In the previously used example of google.com, 'google' would be the name followed by TLD of 'com' separated by the period. Recently there has been a large number of additional TLDs that have become available for public use such as .club or .me which give additional personality for users, allowing for domains such as google.me to be possible.

1. Domain Name System (DNS)

1.1 Utilization

When a client wants to resolve a particular address (we will continue with the example of google.com), it petitions the DNS server with a query to receive the appropriate information. The handling of the query is done by a DNS resolver that is client-side and is responsible for initiating the recursive or non-recursive query to the DNS server to resolve the domain name into an IP address. A non-recursive query provides a DNS record for a domain which it has ownership over or provides a partial result. A recursive query may query other servers in a recursive manner as needed until it has all of the information required. Most DNS servers will issue a recursive query to find out the information requested about the particular domain, asking other DNS servers in the form of a DNS forwarder for the queried information[1].

DNS has additional uses other than just providing translation services for domain names into IP addresses for devices.

The DNS server's files can be queried to find other relevant information about the records of a particular domain. Continuing with the example of google.com, mailing systems utilize MX (mail exchange) records to correctly forward email for a particular email address on that domain. DNS servers are also commonly utilized for the creation and distribution of blacklists for spam emails and other services such that a easy to remember domain could be queried to find out if a particular host is a member of the blacklist. If a particular host is indeed a member of the blacklist then often the host is denied connection. Another example of how DNS has a variety of uses is utilization of DDNS (Dynamic DNS) to update DNS entries in the case of a dynamic IP address which frequently changes. DDNS is useful if a particular host device needs to be accessed at a consistent location, but the IP address that is tied to it is constantly changing. With DDNS the changing of the IP address can be compensated for as a result of the DNS system being updated with the new information as it changes. It is possible to poison or spoof the DNS records that are returned to a host, diverting traffic away from its intended target and often to an attacker's device. For DNS to be functional, there must be a DNS server running to handle the protocol.

1.2 DNS Software

The BIND (Berkeley Internet Name Domain) package was created at the University of California at Berkeley[2]. BIND is an open source software that implements DNS protocols. BIND is by far the most widely used DNS software on the Internet and is currently in use by everyone ranging from government agencies to hobbyists. The BIND package is so popular because it allows authoritative functionality, recursive functionality, access control, caching features, IPv6 support, is open sourced, wild-card support and split horizon support[2]. BIND can be used on both private and public networks alike without considerable changes to the configuration process, making it very simple to setup and get started using. BIND has taken some precautions against DNS spoofing attacks on their servers. Implementation has been made on BIND servers running version 9.5 and above to ignore DNS records that are passed back to the server without a relevant query for them[3]. Other DNS servers also are available that are not as prevalent such as Dnsmasq (a more lightweight alternative to BIND) or djbdns. DNS servers are commonly susceptible to DNS spoofing, creating issues in trust between the client and the server.

2. DNS Spoofing

DNS Spoofing differs from ARP Spoofing in that it replies to DNS requests with falsified information as opposed to sending false ARP requests to impersonate a router. As noted in Man in the Middle (MITM) ARP Cache Poisoning Explained article, ARP packets are considered to reside within the second layer of the OSI model[4]. The second layer of the OSI model represents the data link layer. ARP packets are never routed meaning it is only meaningful in network situations where the devices involved in the ARP communication are locally visible to each other. The third layer of the OSI model represents the network layer; DNS packets are considered to be part of layer 3 because they are able to be routed. The DNS packets are able to be spoofed from any location, not just locally visible locations such as the case of ARP packets. The possible remote nature of DNS spoofing makes it much more of a threat in comparison with ARP spoofing which must be done internally.

2.1 DNS Spoofing Process

DNS spoofing is carried out by replying back to a DNS server with a DNS packet with falsified information. The process allows an attacker to make changes to a DNS entry so that DNS entry is directed to an IP of the attacker's choosing. Originally, BIND was configured such that DNS packets produced were incremented by one for each query that was produced. The easy to predict the nature of the query ID allows an attacker to pose as a legitimate response to that query before the actual response comes back, potentially allowing the storage of incorrect information in DNS records. The other piece of information that would generally be required is the port at which the DNS transactions were taking place. The port that is used by BIND and most other DNS software is generally consistent and acting on a particular port and the port is reused. As a result of the almost universal use of BIND, figuring out that the port number is not much of a hurdle for an attacker to figure out. To find the source port, an attacker would only need to issue a lookup request through the DNS server. Upon receiving the valid information from the server, the port at which the BIND server communicated on can be identified easily as the source port. The source port for that transaction that was just carried out is not randomized by BIND and as a result, it does not do itself any service in thwarting attacks in this manner. An attacker generally would bombard the DNS server with a large amount of DNS packets with a large TTL (Time to Live). The TTL represents the amount of time that the data that is received in the packet is thought to last before needing to be refreshed. Generally, TTL is used to improve on system performance by caching certain information such that the same lookups do not have to be done over and over. After a query is made by the DNS server and a record is received back, the TTL specifies the amount of time that that record will remain valid before having to be rechecked. When a request is made for google.com for example, if the DNS server has a cached record for that DNS, it will provide the client that information as opposed to starting a new query to translate the domain every time. The ability to store this information greatly increases the performance of the server and allows lesser loads on nameservers. If an organization has multiple users and user A makes a query for google.com and that record has not previously been stored in the cache or the TTL of the previous record has expired, that record will be queried for. When user B attempts to reach the same domain name, the cached information will be supplied as opposed to

starting a new query.

Generally, when making modifications to DNS records, an administrator will change the TTL on certain information such as MX (Mail Exchange) records to lower values such that when the change is applied it is more quickly refreshed and the correct information is applied. At the time of introduction of BIND and other DNS software, the common value for a TTL of a DNS packet was about 24 hours. The common value for a TTL for such packets has gone down as a result of realization of possibility of malicious activities as a result of the high TTL time from DNS spoofing and cache poisoning. A lowered value of TTL does not solve the issue and DNS spoofing and cache poisoning attacks are still possible. If a DNS server has a poisoned cache, an attacker would be able to control the translation processes from hostname to IP address and effectively control the traffic of those devices, directing them to wherever they please, regardless of the validity of the record. The vulnerabilities that are seen in the DNS system can be directly attributed to the lack of any sort of authentication mechanisms in the DNS system.

2.2 Birthday Attack

A popular method of an attack on DNS servers is often referred to as the birthday attack. The birthday attack is based on the birthday problem and subsequent birthday paradox, first discussed by W.W. Ball in relation to mathematical probability and statistics[5]. The birthday problem deals with the problem of finding the percentage probability that a pair of people in a set of randomly chosen people will have the same birthday[6]. There are 366 possible birthdays for an individual, counting the possibility of a leap year ; as a result the probability of a pair of people will share the same birthday would be 100 percent when the number of people included in the study is greater than 366. The birthday paradox with this problem is that in order to reach a 99.9 percent probability rate that there would indeed be a pair of individuals in the set with the same birthday, only 70 people will need to be included; even more interesting is that the probability of a pair of people sharing the same birthday is just over 50 percent when only 23 people are included in the study.

A method of DNS poisoning would be to bombard a DNS server with responses that are crafted by the attacker. Protocols in the DNS system are already in place that require the responses to include the ID of the request of which it is a response as well as the port on which the transaction took place on. As previously discussed, the issue of the attacker to figure out the communication port is trivial and often a non-issue for an intent attacker. A birthday attack is carried out by an attacker by causing the DNS resolver to “issue multiple queries for the same domain in order to increase the probability of a match” with one of the fake responses that have been generated by the attacker[7]. In this manner, a birthday attack is a very efficient brute force type of attack that can be easily applied with minimal resources. In the case of a birthday attack on a DNS server, the same methodology can be applied as seen in the birthday paradox, but instead of the limit of possible birthdays being 366, the possible limit of transaction IDs that are produced by the DNS server is 65535[6].

In application, the birthday attack is a very powerful tool in spoofing DNS records as it is able to reach a 50 percent probability rate of being successful in only 300 attempts. Each attempt in our application would be a response to the DNS server with forged information, attempting to get lucky and forge the correct transaction ID such that the DNS server accepts it. A 99.9 percent success rate is found in the 600-700 packet range, meaning the time required for a successful attack is very minimal using this method[7].

The time at which the real nameserver replies back with the legitimate information is the only hurdle that is left for an attacker utilizing the birthday attack to overcome. The attacker must ensure that the correct packet that is part of the batch of packets that are being bombarded to the DNS server reaches the server before the legitimate packet from the real nameserver is received. Because of the speed at which the birthday attack is able to narrow down the possibilities, the amount of time required for this process is minimal, but an attacker could utilize methods such as flooding to slow down the server to ensure that the attacking packets get there first[6]. Once the attacker beats the real nameserver in the reply, the attack is complete for the duration of the TTL that was specified by the attacker when creating the falsified response that was sent to the target DNS server.

3. BIND

One of the most important parts about the BIND system is that the transaction ID needs to be matched for the response to be accepted. A tool has been created that does analysis of the random number generation processes. The tool was applied to BIND 8 and it was found that the random number generation process was not an ideal random number generator, meaning some numbers had a far higher likelihood of being selected in comparison with others (Stewart, 2003). An algorithm was created, proving that, when provided with the previous three transaction IDs, it would be possible for an attacker to guess with a probability of 100 percent the next transaction ID that is to be generated based on the random number generation process within BIND 8. Changes as a result of this issue with the random number generation within BIND 8 were made in the subsequent release of BIND, slightly alleviating this issue. In BIND 9, the random number sequence that is generated is a little more secure. Instead of being able to guess the next transaction ID with 100 percent certainty with only three previous transaction IDs, BIND 9 solves some of the problems and lowers the certainty to predicting a random number to be generated to 20 percent while having access to 5,000 previous transaction IDs[8]. While this random number generation process in BIND 9 is still not good enough as it is still somewhat predictable (and as a result, susceptible) as to the random numbers that are being generated, the percentage likelihood is greatly reduced in comparison with previous

versions[6]. Djbdns's random number generation is slightly worse than that of BIND 9, allowing a predictability of 30 percent on the same sample size, but because the source ports are also randomized every time in addition to the transaction ID, it would take billions of combinations to correctly match a query[6]. Microsoft DNS server is another commonly used DNS software that is used in enterprise environments that is equally susceptible to DNS spoofing as BIND as a result of the fixed source port. Microsoft DNS server keeps the source port for queries fixed and as a result the attacker does not need to make attempts to guess the source port. The advantages shown through djbdns and the randomization of the source port can be seen in the ability to effectively provide a large amount of protection just from the extra randomization.

3.1 Post-Attack

Once an attacker has become successful with a DNS spoofing attack, the attacker has full control over web traffic and is able to launch man in the middle (MITM) type attacks. An attacker could redirect the users of the network to phishing sites that trick users into entering sensitive information, thinking they are visiting a legitimate site, allowing the attackers access to their credentials.

4. Other DNS Spoofing Applications

More and more people are switching to use VoIP (Voice over Internet Protocol) services for their home and cellular phone needs. DNS spoofing can be used by an attacker to exploit the VoIP protocols and cause havoc on a user's VoIP service. A case study investigated the possibility of attacking Vonage service VoIP phones utilizing DNS spoofing, showing the feasibility of such an attack is much more effective than previously thought[9]. An attacker, possibly through some of the methods previously discussed, could redirect traffic through their own SIP server by sending spoofed DNS responses back to the phone. All of the calls that are made through the Vonage phone would then be passed through the attacker, at which point the attacker could "wiretap and hijack any calls to or from the targeted phone"[9]. The implications of properly securing the DNS server and ensuring the DNS server does not have a poisoned cache has severe implications. As technology grows, DNS will continue to be depended on by various devices and services to properly route information. Ensuring DNS servers are properly configured and secured such that the possibility of attack is minimized should be at the forefront of efforts as often it is overlooked and utilized by attackers as an easy means of access to a network.

5. Keeping up to Date

Ensuring that DNS software is fully updated to the most recent versions will allow an organization to assist to ensure that issues that are found are patched on the machine running the DNS server. Although the process of patching software is sometimes tedious, the benefits of updating to the most recent version of BIND or other software are far greater. Often it is not possible to upgrade the DNS software, for example, in the case of government regulations where a particular set of software is supposed to be running on the server and upgrading or manipulating that is not allowed. Many servers still run older versions of BIND that are susceptible to the types of DNS attacks that have been talked about previously. Some of the exploits have since been patched or severely reduced in more recent versions, lingering on outdated software versions opens up the network environment to possible exploits and attackers. In more recent versions of BIND, a credibility indicator has been included to identify the credibility of the source of the data. When BIND detects information from a higher credible source differing from what it has already stored, it will refresh the information with the information gained from the highest credible source that is possible[10]. While still possible to launch a DNS attack on an updated BIND server, the amount of exploits is greatly reduced.

6. Conclusion

DNS spoofing and other exploits on the DNS system are commonly used by attackers to gain access to sensitive information. DNS spoofing should be considered a real threat in a network environment and precautions should be taken to best mitigate possible exploits relating to DNS. Significant strides have been made recently in properly securing popular DNS software and other loopholes that are commonly used by attackers. Secure systems should not rely on DNS. Typically on highly secured systems, access to the internet is limited to begin with reducing the need to use DNS altogether on those machines. By not using DNS on secure systems you in effect reduce the possible exploitation of the DNS protocol[11]. Constant monitoring of network activity should be a standard in assisting to ensure that even a properly secured network is not falling prey to a DNS based attack.

Acknowledgments

This section needs to be redone.

References

[1] S. Hanley. Dns overview with a discussion of dns spoofing, 2000.

[2] BIND. Internet systems consortium. Web, jan 2014. https://www.isc.org/downloads/bind/.

[3] S. Graves. Bind 9 security vulnerability matrix. *ISC Knowledge Base*, may 2014. https://kb.isc.org/article/AA-00913/0/BIND-9-Security-Vulnerability-Matrix.html.

[4] N. Briscoe. Understanding the osi 7-layer model. *PC Network Advisor*, 2(120), jan 2000.

[5] R. W. W. Ball. Mathematical recreations and essays, jan 1914. MacMillan.

[6] J. Stewart. Dns cache poisoning–the next generation. *Secure Works*, aug 2003. http://www. secureworks. com/research/articles/dns-cache-poisoning.

[7] A. Herzberg and H. Shulman. Security of patched dns. *Computer Security–ESORICS 2012*, 1:271–288, jun 2012. Springer Berlin Heidelberg.

[8] A. Klein. Bind 8 dns cache poisoning. 1, sep 2007.

[9] R. Zhang, X. Wang, R. Farley, X. Yang, and X. Jiang. On the feasibility of launching the man-in-the-middle attacks on voip from remote attackers. *roceedings of the 4th International Symposium on Information, Computer, and Communications Security*, pages 61–69, oct 2009. ACM.

[10] P. Vixie. Dns and bind security issues. *Proceedings of the 5th USENIX UNIX Security Symposium*, jun 1995. USENIX Association, Berkeley, CA.

[11] C. Sanders. Understanding man-in-the-middle attacks – part2: Dns spoofing. *WindowSecurity.com*, aug 2010. http://www.windowsecurity.com/articles-tutorials/authentication-and-encryption/Understanding-Man-in-the-Middle-Attacks-ARP-Part2.html.