A Novel Approach - Effective Method of Prevention of Cache Poisoning for Wild Card Secure DNS

K.Sampath Kumar [#1] Dr.G.K.D.Prasanna Venkatesan [*2] R.Vignesh [#3]
[#] Computer Science and Engineering
[#] PGP College of Engineering and Technology, Namakkal, Tamil Nadu, India
[#1] ksampathkumara@yahoo.com
[#3] tharunvignesh120611@gmail.com
[*2] Professor & Head
[*2] Electronics and Communication Engineering
[*] PGP College of Engineering and Technology, Namakkal,Tamil Nadu, India
[*2] prasphd@gmail.com

Abstract - **Recursive DNS (RDNS) resolvers on new attack for poisoning the cache was discovered in DNS. DNS software only partially protected the DNS poisoning in servers. In this paper we discuss two type of DNS poisoning prevention methods 1) wild-card secure DNS (WSECDNS) and 2) Start of authority (SOA) in without changing protocol. In wild-card given in the RFC 1034, RFC 4592 and TXT resource records in order to increase the flow of DNS queries to the point that cache poisoning attack infeasible. The stub-resolvers depend on the RDNS for the IP address. It is responsible for the direct contraction authoritative name servers on stub-resolvers and the cache responses for a given TTL (Time to Live) and then forwarded it back to stub-resolvers. Now the poisoning attack works by forcing an RDNS to look up a domain name and then sending a forged before RDNS gives back the domain to the server. As per in this paper the RDNS sends authoritative name to the server it provides and recommends a TLD(Top Level Domain) server as we don't give any knowledge of the IP address to the RDNS. Now RDNS will ask for the TLD servers for the IP of authoritative name server. TLD will response with the recommend to SOA (Start of Authority) for the domain and finally gives record to stub-resolvers through the path of RDNS. So the attack over the cache is not possible as the domain is sent from the SOA.**

Keywords- ***RDNS, WSECDNS, TLD, TTL, SOA***

I. INTRODUCTION OF DNS

The Domain Name System servers in our everyday life to provide us with the correct domain name to IP address mapping, so that we can browse the web, send emails, access emails, access our bank accounts, Booking tickets and etc In the world of the Internet and TCP/IP , IP addresses are used to route packets from the source to destination. A single IP address, for example 009.009.009.009, is not difficult to remember. But trying to learn or track thousands of these addresses, including which server/node is associated with each address, is a threatening task. So we use domain names to refer to systems with which we want to communicate. In real world Internet domain name example is Google.com. When you enter the Google domain name into the address bar of browser, the Google page appears. The PC executed a process to resolve Google.com to an IP address. The IP address is a system able to initiate a session with another system across the Internet. This is two ways IP address resolution function can occur. The domain name / IP address resolution process when the target system and DNS server are internal [1]. A workstation must establish a session with a server with a domain name of Google.com and also workstation to implement DNS; it must be running a DNS Client or Client Resolver. The Background of DNS which consists of the following ingredients.

a) Stub resolver: The originator of a DNS query. This could be a simple client machine or a Web browser.

b) Recursive DNS (RDNS): This is a server machine that is responsible to assist the stub resolver on resolving a domain name. These servers maintain a local cache of past resolved domain names for a certain period of time to live and are the main target of cache poisoning attacks.

A. Role and Function of DNS

DNS queries are usually initiated by a stub-resolver (Ex., a Web Browser) on a user's machine, which depends on a recursive DNS resolver (RDNS) for obtaining the IP address (or other Resources) related to a domain name. The RDNS is responsible for directly contacting the authoritative name servers on behalf or the stub-resolver, cache the response for a given time to live (TTL) and forwards it back to stub-resolver [4].

The Basics steps for Resolver is following:

- Step 1: The resolver checks the resolver cache in the workstation's memory to set the contains an entry for Google.com.
- Step 2: Having found no entry in the resolver cache, the resolver sends a resolution query to the internal DNS server
- Step 3: When the DNS server receives the query, it first checks to see if it's authoritative for the Google.com domain. The server performs a lookup in its internal zone table and also hosts Resource Record (RR) that includes the IP address for Google.com.

- Step 4 : The IP address of Google.com is returned to the resolver.
- Step 5: The resoled domain name and IP address are placed into the resolver cache.
- Step 6:Hence the end user is been send reply for www.google.com

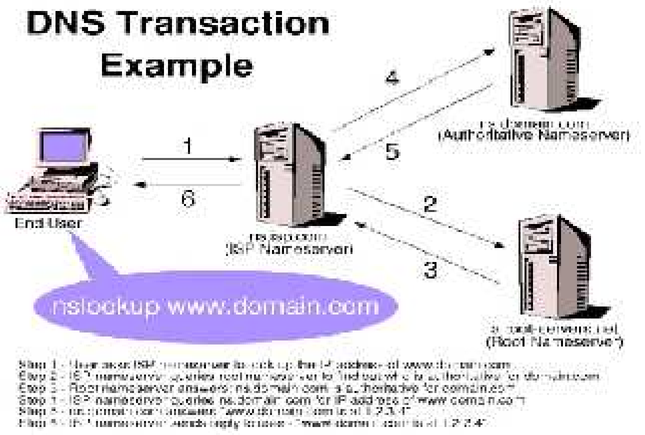

Figure1: Basics producer for Resolver

II. DNS CACHE POISONING

DNS cache poisoning consists of changing or adding records in the resolver caches, either on the client or server, so that a DNS query for a domain returns an IP address for an attacker's domain instead of the intended domain [5]. DNS has been found to be vulnerable to a number of attacks. In particular, cache poisoning attacks have been shown to be quite feasible [2]. Poisoning attacks work by forcing an RDNS to lookup a domain name (Google.com) and then sending forged DNS responses back to the RDNS before the real valid response from an authoritative name server arrives. Each DNS query contains a 16 -bits long transaction ID (TXID) that allows the RDNS to distinguish valid responses from bogus ones. Therefore the attacker has to guess the correct TXID in order for a forged response to be accepted and stored in the cache. If the attack is successful the attacker can force the RDNS to resolve the targeted domain name to a malicious IP, and to store the malicious IP in the cache with a long TTL. The DNS cache poisoning recover methods are 1. RDNS approach model 2. MSEC DNS approach model. 3. DNS Cache poisoning without protocol model. In this paper we discuss and case study about three types of DNS cache poisoning approach models [1].

A. RDNS approach Model

A new attack for poisoning the cache of Recursive DNS (RDNS) resolvers was discovered and revealed to the public. In major DNS vendors released a patch to their software. However, the released patch does not completely protect DNS servers from cache poisoning attacks in a number of practical scenarios. DNSSEC seems to offer a definitive solution to the vulnerabilities of the DNS protocol [6]. The implementation and deployment of DNSSEC would therefore provide a robust way of protecting against DNS cache poisoning attacks because all the responses are signed and their authenticity can be verified. For example, DNS cache poisoning attacks would not work because forged responses can be identified and discarded. DNSSEC seems to be the panacea for the vulnerabilities of DNS. In generally DNSSEC is not enough too adopted and deployed in a large scale. It is based on the current DNS protocol and work by increasing the entropy of DNS queries in order to make forging a valid response more difficult. A novel solution to brute-force DNS cache poisoning attacks that is based on increasing the entropy of DNS queries to the point that cache poisoning attacks become practically infeasible.

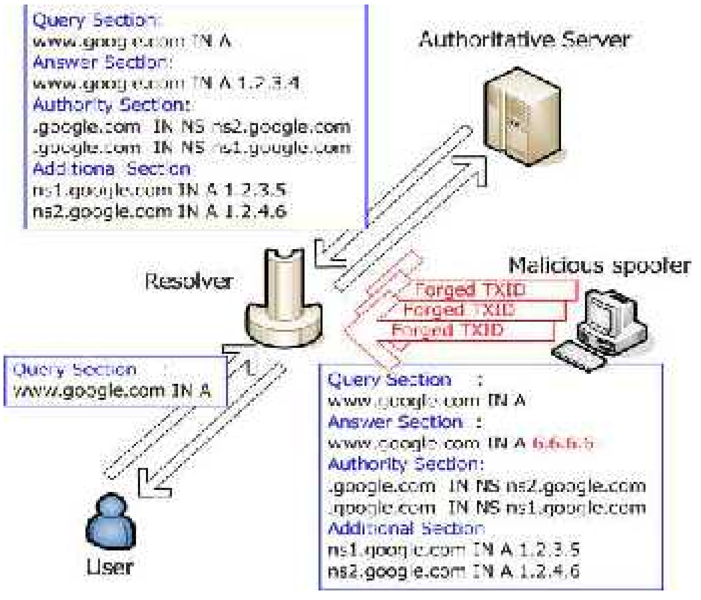
Query Section:
www.google.com IN A
Answer Section:
www.google.com IN A 1.2.3.4
Authority Section:
.google.com IN NS ns2.google.com
.google.com IN NS ns1.google.com
Additional Section
ns1.google.com IN A 1.2.3.5
ns2.google.com IN A 1.2.4.6
Authoritative Server
Resolver
Malicious spoofer
Forged TXID
Forged TXID
Forged TXID
Query Section :
www.google.com IN A
Query Section :
www.google.com IN A
Answer Section :
www.google.com IN A 6.6.6.6
Authority Section:
.google.com IN NS ns2.google.com
.google.com IN NS ns1.google.com
Additional Section
ns1.google.com IN A 1.2.3.5
ns2.google.com IN A 1.2.4.6
User

Figure 2: DNS cache poisoning

B. WSECDNS approach Model

In Wild-Card Secure DNS (WSECDNS), a novel solution to DNS cache poisoning attacks WSEC DNS relies on existing properties of the DNS protocol and is based on wild-card domain names. WSEC DNS is able to decrease the probability of success of cache poisoning attacks by several orders of magnitude [6]. A new DNS query process that leverages existing properties of the DNS protocol. The wildcard domain names given in RFC 1034 [9] and RFC 4592 [10] and TXT resource records and random strings have the effect of significantly increasing the entropy of DNS queries, thus making valid answers difficult to guess (eg. *. Google.com).A wild card domain name is a domain name having its initial (i.e., leftmost or least significant) label be the "*" character [10]. For example *.www.google.com is the valid domain, where"*" is interpreted as "any valid combination of characters". WSEC DNS guarantees complete backward compatibility with current DNS (RDNS) resolvers that intend to take advantages of the security benefits of WSEC DNS must execute some new functionalities. It provides a way to protect RDNS resolvers from brute-force cache poisoning attacks including kaminshy's attack [8].

The background of a traditional brute-force DNS cache poisoning attack scenario is as follows:

- Assume an attacker tries to poison the IP address of www.Google.com. The attacker first sends a query for www.Google.com to the RDNS and interacts with the authoritative name servers.
- If the attacker is able to guess the TXID and source UDP port and send well crafted response packets to the RDNS before the legitimate answer from the real authoritative name server is received, the DNS poisoning attack will be successful. This attack works because the RDNS will accept the first valid answer it receives. As a result, it will store the IP address that the attacker sent in the positive cache for the entire time to live chosen by the attacker[1].

For example, every time the users want to visit a web page on that domain, they may be redirected to the attacker's malicious website. This may expose the users to a variety of attacks such as information theft or malware inflection. The benefits of WSEC DNS protect the RDNS's cache from poisoning attacks against the domain names in WSEC enabled zones, including Kaminsky's attack [8]. If the attacker is not able to forge packets with the correct combination and send them to the RDNS before the genuine authorititative response arrives from the ANS, the attack will fail. The transparency property of the WSEC DNS query process is completely transparent to the host thanks to the WSEC response normalization algorithm and independently from the RR type requested in the original query from the host, thanks to the use of CNAME wildcards added in order to make a Zone. The WSEC TXT resource records were originally meant for storing descriptive text about domain names, but are now widely used to carry information related to the sender policy framework to mitigate the spam emails phenomenon.

C. Implementation of WSEC DNS

Implementing the WSEC DNS query process without a WSEC caching system would have the side effect of doubling the volume of DNS traffic on the Internet and the average latency between users DNS queries and the related answer, due to the fact the handshake between the RDNS and ANS would have to be repeated for each query. An entry of the WSEC positive cache should contain the following information: a) a zone name and b) a time to live, after which a TXT query for the WSEC handshake needs to be reissued. Once a zone has beeb stored in the positive WSEC cache the RDNS will not perform the WSEC handshake for domains in that zone until the TTL of positive WSEC cache expires. An entry in the WSEC negative cache should contain two information i) the name of the non-WSEC enabled zone 2) the negative entry TTL.

The probability of successful cache poisoning for one attack can be computed as

$$p_{succ} = 1 - p_{fail} = 1 - \prod_{i=0}^{M-1} (1 - 1/\Gamma - 1)$$

$$\Gamma > M - 1$$

Where n spoofed DNS answer to the RDNS server within the RT and also n depends on the bandwidth BW available to the attacker. Γ represents the overall cardinality of search space. The total probability of success after launching n instances of the attack can be computed as

$$p_{succ} = 1 - (1 - p_{succ})^{n}$$

WSEC DNS provides complete backward compatibility with name servers that do not intend to support WSEC DNS queries, thus allowing for an incremental deployment. This method is root and top-level-domain (TTL) approach with large scale in short period of time.

D. DNS Cache poisoning without protocol model

In this model DNS using SOA functions. (ie) A type of resource record that is used by the Domain Name System (DNS). Every domain name has an SOA record in its database that indicates basic properties of the domain and the zone that the domain is in. The SOA record contains: The host name for the primary name server for the zone. An e-mail address of the person who is responsible for the domain. The serial number for the zone. The refresh interval. This is how often, in seconds, the secondary name servers check with the primary name server to see if any changes have been made to the domain's zone file [7]. The time, in seconds, a secondary server waits before retrying a failed zone transfer. This time is typically less than the refresh interval. The time, in seconds, before a secondary server stops responding to queries, or "expires" a zone, after a lapsed refresh interval where the zone was not refreshed or updated. The minimum time-to-live (TTL). This value is supplied in query responses by servers for the zone to inform others how long they should cache a resource record provided in an answer [7]. The DNS cache poisoning without protocol includes the following fact all along with stub-resolver and recursive DNS steps.

a) Root and Top Level Domain (TLD) Servers: These are servers that provide referrals to the TLD's and Start of Authority servers respectively.

b) Start Of Authority (SOA) : The SOA servers represent the authoritative name server for an entire name zone.

Basic Function of SOA as follows:

Step 1 & step 2: Includes the general base lines that take place in the ordinary DNS system process.
Step 3: Now the SOA provides the recommend to the TLD Servers.
Step4: RDNS will ask the TLD Servers for the IP of Google.com
Step 5: TLD will response with a recommend to the SOA for the required domain.
Step 6 & Step 7: RDNS will finally get the A record containing the IP address for Google.com.
Step 8: RDNS will forward the answer to the sub resolver.

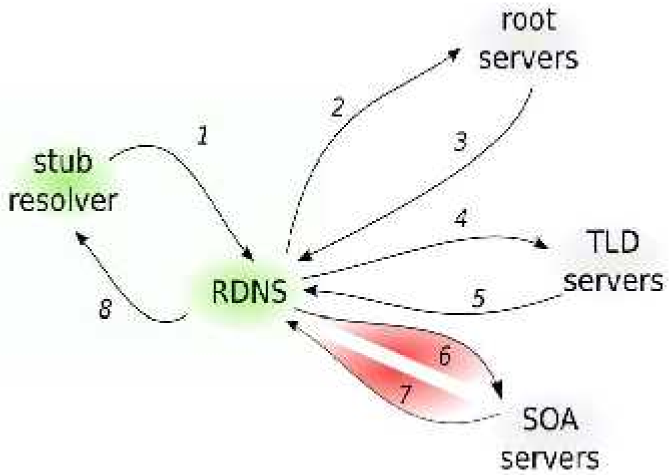

Figure 3: Basic producer of SOA

The RDNS normally to verify the following information's

a. The queried domain name has to match the domain name reported in the question section of the answer.
b. The TID of the answer has to match the TID in the query issued by the RDNS server.
c. The destionation IP and source UDP port used by the RDNS server for sending the request have to match with the source IP and destination UDP port on which the answer is received.

III. ADVANTAGE OF WSECDNS & WITHOUT PROTOCOL MODELS

1. No change to the DNS protocols.
2. No software change for root and the domain server.
3. No complication in DNS traffic volume.
4. Transparent to users.
5. Poisoning attacks are practically infeasible.

IV. CONCULSION

The implemented of WSEC DNS would certainly introduce some overhead on the RDNS servers that implemented it. These WSEC DNS is considered in performs with concern, only for short domain names (e.g. google.com, yahoo.com).On the other hand the SOA provides enough protection towards the poisoning attack for longer domain names (.e.g. as long as 16 characters, or longer).The process of using WSEC DNS only for relatively short domain names would alleviate the computational over head and cache related memory consumption for RDNS servers deriving from the WSEC query process.

REFERENCES

[1] Roberto Perdisci, Manos Antonakakis, Xiapu Luo and Wenke Lee Damballa, IEEE/IFIP DSN-DCCS 2008.

[2] J.Stewart,DNS cache poisoning - the next generation http://www.secureworks.com/research/articles/dns-cache-poisioning/ 2002.

[3] Tom Olzak , http://technet2microsoft.com/WindowsServer/en/Library Mar 14,2006.

[4] http://blbaliyase.blogspot.in/2009/11/dns-cache-poisioning.html, Nov 26 2009.

[5] Hyatt,R.(2006,January).keeping DNS trustworthy.the ISSA Journal. January 2006.

[6] Perdisici, R. Dependable Systems and Networks ,IEEE/IFIP, July 2 2009

[7] http://www.webopedia.com/TERMS/S/Start_of_Authority.html

[8] D.Kaminsky, Presented at BlackHat 2008.

[9] P.Mockapetris. Domain names – concepts and facilities, November 1987. http://www.ietf.org/rfc/rfc1034.txt.

[10] E.Lewis.role of wild cards in the domain name system, July 2006 http://www.ietf.org/rfc/rfc4592.txt.

AUTHORS PROFILE

1) K.Sampath Kumar, Assistant Professor / Computer Science and Engineering Department at PGP College of Engineering and Technology, Namakkal, Tamilnadu, India. Presently Research Scholar in Anna University, Chennai, India.

2) Dr.G.K.D.Prasanna Venkatesan, Completed Ph.D from College of Engineering, Anna University, Chennai,India. He is Currently Working as Vice-Principal, Professor & Head of Department of Electronics and Commuination Engineering at PGP College of Engineering and Technology.Namakkal, Tamilnadu,India. His research interests includes Wireless Sensor Networks, 4G Wireless Networks, Cloud Computing, adhoc Network, etc.,

3) R.Vignesh, Student of Bachalor of Engineering in Computer Science and Engineering at PGP College of Engineering and Technology,Namakkal.