Security Issues with Self-Signed SSL Certificates

Rishabh Kumar, Ashwin Perti

Abstract: ***The price difference between self-signed SSL Certificates and certificate issued and verified by a trusted third party is the main reason for using a self-signed SSL Certificate. What organizations miss is the security threats associated with the self-signed certificates and self-signed certificates may be pricey in the long run. SSL security is the single most important authentication protocol used in web-based transactions. While organizations continue to rely on SSL security for external facing sites such as a company home page or e-commerce sites, some IT professionals believe that self-signed SSL certificates are an acceptable alternative for internal sites. Because only internal employees have access to servers that host internal-facing sites including corporate email servers, human resource portals, and wikis, for example, these IT specialists believe self-signed SSL certificates provide adequate protection at a lower price point. This paper aims to highlight security issues associated with self-signed SSL certificates and how those threats can compromise the security of the system. In this paper, a detailed methodology on how to perform a man-in-the-middle attack on self-signed SSL certificate is also given and code to reproduce the attack is publicly available on git repository (given in section 3). We conclude the paper by recommending some measures to prevent such attacks and enhance the security of the system.***

Keywords: SSL; certificates; Self-Signed Certificate; Man-in-the-middle attack; ARP Poisoning; DNS Poisoning; SSL security ; usability.

I. INTRODUCTION

Providing security so that end user can communicate sensitive data online was a significant achievement in the development of web and an essential condition for its growth. While the functionality of web has changed significantly from serving static HTML pages to providing the platform for the commerce of billions of dollar the core SSL/TLS technology persists as the basis for securing many aspects of today's Internet including software download, data transfer, user passwords, and for site authentication.[1] SSL protocol aims at securing HTTP protocol and the combination is HTTPS (HTTP over SSL/TLS) where S stands for secure. There is no doubt that SSL/TLS encryption provide protection against the majority of cyber threats but self-signed certificate may result in a false sense of security and is as good as no security. One can not be blithe by merely using a self-signed SSL certificate and MITM attacks are very much feasible on the web service. This paper aims to highlight security issues associated with a self-signed SSL certificate and provides a methodology to perform a Man-in-The-Middle attack by generating certificates on the fly. The following paper is divided into five sections:-

1) Literature Survey
2)Background of SSL.
3) Threats associated with self-signed SSL.
4) Methodology to perform a Man-In-The-Middle attack.
5) Recommendations.

II. BACKGROUND OF SSL

The main purpose of SSL is to provide a secure communication channel between two peers, acting as a technology that facilitates secure key exchange, encryption, authentication, and integrity. SSL was designed to be resilient against attacks like replay attack, eavesdropping, session hijacking, and man-in-the-middle attacks. Although SSL has the capability to authenticate both the parties in communication, in general, it is only the server that authenticates itself. SSL also aims to provide interoperability and extensibility in addition to security.

SSL is divided into two layers: the record layer and handshake layer. The task of the record layer is to handle the data provided by the application layer. The data is fragmented into blocks, further compression is applied to reduce the size, and symmetric key encryption with MAC digest is generated at last. The keys used by the record layer in symmetric key encryption are predetermined by the handshake layer on per session basis. The goal of the handshake layer is to establish a session and perform operation negotiation in addition to determining keys for the record layer.

In reference to TCP/IP model, SSL/TLS works "above"(refer Fig-1) the transport layer (TCP/IP protocol), which is responsible for process to process communication over the internet and "below" application layer (HTTP protocol). This implies that in the SSL enabled communication SSL uses TCP/IP on behalf of the application layer protocols.

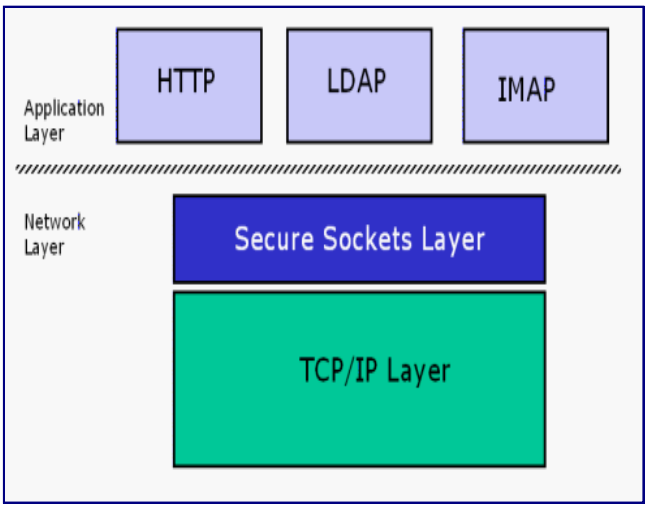

Fig. 1 Location of SSL[11]

Revised Manuscript Received on May 28, 2019.

Rishabh Kumar, IT Department, ABES Engineering College, Affiliated to Dr. A.P.J. Abdul Kalam Technical University, Ghaziabad, India
Ashwin Perti, IT Department, ABES Engineering College, Affiliated to Dr. A.P.J. Abdul Kalam Technical University, Ghaziabad, India

A typical SSL transaction involves two phases:-
1) Handshake Phase.
2) Data Transfer Phase.

In the handshake phase both the server determine the secret-key parameters by using asymmetric key encryption. In the data transfer phase, the secret key is used to encrypt and decrypt the message. The connection is initiated by the client by first sending a Hello message to the server. The hello message includes cipher specs which are a list of secret-key algorithms supported by the client. Server reply to the client by a similar Hello message, also it provides the name of cipher spec chosen by the server for further key exchange. Succeeding the Hello message, the certificate sent by the server. A certificate serves the purpose of authenticating the identity of the server and contains the public key of the server, server's ID, and several other specifications. The server gets this certificate from a trusted third party known as the Certificate authority. A secure channel is used by the server to send its public key to the CA. The certificate generated by CA contains its own identification number (to identify the CA issuing certificate), server identification number, the public key of the server, and several other parameters. The CA after generating the certificate uses a message digest to create a certificate fingerprint. A fixed-length output is generated by the message digest taking variable length input. Certificate signature is then created by CA using fingerprint and encrypting it by its own private key. When the client receives this certificate from the server, it first deciphers the signature using the public key of CA and read the pre-calculated fingerprint. After that client computes the fingerprint of the certificate using a message digest algorithm. This helps to determine the integrity of the certificate and if the two fingerprints don't match then the certificate has been tampered with. To verify that CA issuing the certificate is trusted or not client maintains a list of trusted CAs along with their public key. Once the server's authentication is complete, asymmetric key algorithms can be used to deduce secret-key information. This completes the handshake phase and the client and server transition into data transfer phase where the data is exchanged securely.

III. LITERATURE SURVEY

In this work we have studied the security issues associated with Self-signed certificates.

1) In the [1] author has discussed about the security of network in general. He has highlighted various threats to secure communication in a network. In our paper we have focussed on SSL based communication and discussed about security threats associated with using self-signed SSL certificate.

2) In [2] is a article in the book "**Applied Computing and Informatics**" which discusses the security that SSL provide in HTTPS and also highlights that there is a potential threat of Man-in-the-middle attack in SSL. Author talks in depth about how SSL encrypts the transactions and provide security.

3) In the [3] author focusses on the use of mobile devices for communication and how the increasing use of mobile devices increases the complexity to maintain security. Author has done a survey on android SSL implementation and gives us an insight what problems are faced due in android to implement SSL.

4) [4] paper discuss the need of some technique to circumvent SSL encryption. The need to bypass SSL encryption is forensic analysis. In the paper author has given the Proof-Of-Concept of bypassing SSL security in IOS applications.

5) Characterization of cryptographic strength of servers is done in [5]. Author has developed a tool to classify public servers using SSL/TLS into different categories on the basis of cryptographic strength.

6) Mechanism of ARP Poisoning and techniques to detect the attack are discussed in the paper [6]. Our methodolgy to perform MITM on self-signed SSL relies on ARP Poisoning to intercept traffic of the network.

7) DNS serves the purpose of mapping site names to server IP address. Client can be redirected to malicious site through DNS poisoning. In [7] author highlights a DHCP based vulnerability in DNS. Our paper highlight a threat on SSL assuming DNS poisoning as the base attack.

8) In [8] author has developed a protocol to detect man-in-the-middle attack on SSL. The protocol is based on signature generated by CA during certificate issue.

9) In paper [9] author has analyzed the effect of forrged SSL certificate in the system. This analysis gives an insight what potential threats could arise if proper implementation of SSL is not done.

IV. THREATS ASSOCIATED WITH SELF-SIGNED SSL

The less cost is the main reason to use self-signed SSL instead of the CA issued SSL certificate. What the organization doesn't know is the hidden cost and security threats associated with self-signed SSL certificates. One of the purposes that SSL serves is the authentication of the server to the client, but, when we use self-signed certificates the authentication is not possible because the certificate is issued by the organization itself and it like saying "I am verifying that I am me." The trust factor associated with SSL is lost. The browser shows a security warning(Fig-2) when using a self-signed SSL certificate:-

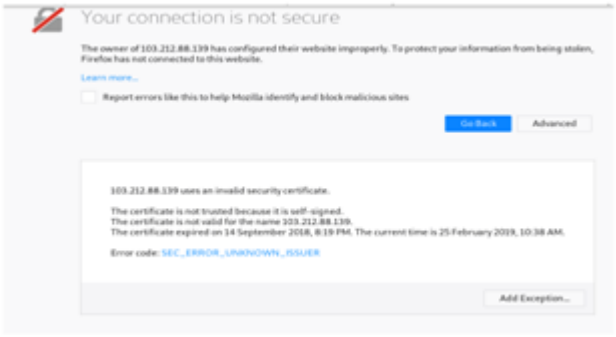

Fig-2 Warning by browser

To browse the site the client must "**Add Exception**" which exposes him/her to many client based attacks. It is very much possible that a man-in-the-middle attack on such a server and client communication will go undetected and any confidential information entered by the client might be stolen by the attacker.

This paper covers the following threats associated with Self-Signed SSL certificates :-
1) Replacing certificate on the fly using man-in-the-middle attack based on ARP poisoning.
2) Using DNS spoofing and redirecting client to malicious site.

Replacing certificate on the fly using Man-In-The-Middle Attack

The elimination of man-in-the-middle attack is one of the main reason for transition from HTTP to HTTPS as it enabled a way of secure transactions between client and server. However when using self-signed SSL certificate, client can not authenticate the server thus become prone to mitm.

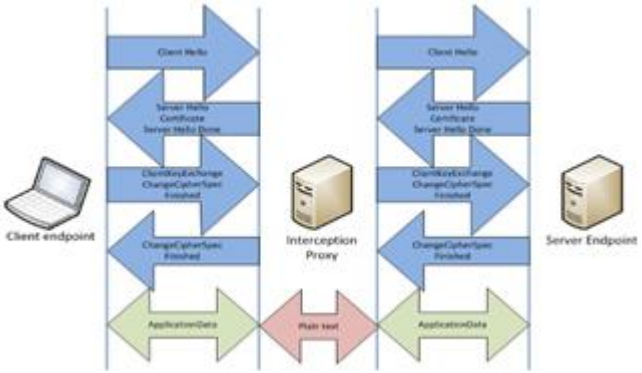

Fig. 3 Intercepting SSL Traffic using MITM[10]

This attack has been described in detail with methodology and code available in section-4 .

DNS Spoofing and redirecting client to malicious site

This attack exploits the vulnerability that in spite of encryption, the client is unable to authenticate the server. In this attack DNS server of the client is spoofed by the attacker and redirects the request of the client to the target site is served by the false reply that redirects him/her towards the malicious site. To understand this attack further let us first understand step by step how DNS works:-
Step-1) Client wants to visit a website say www.xyz.com. To visit the website client must know the IP Address of the site, thus, it makes a request to the local DNS server to resolve the domain name.
Step-2) If local DNS has an entry for the site it will reply to the client giving the IP address of the site. Otherwise, it will further request authoritative DNS to resolve the request.
Step-3) The authoritative DNS reply to local DNS and local DNS forwards the reply to the client. Meanwhile, local DNS maintains a cached copy of reply.

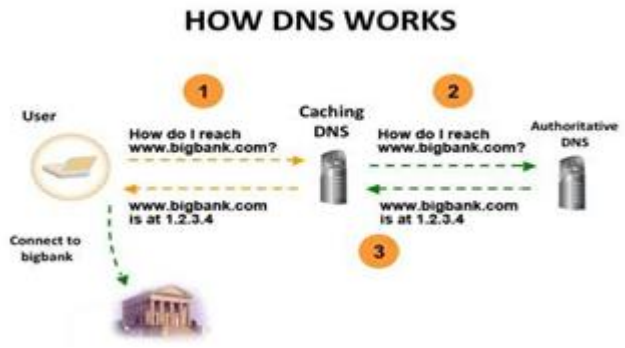

Fig. 4 Working Of DNS[12]

Working of DNS can be exploited to insert fake entries in DNS thereby redirecting the client to desired site. Step by step procedure for DNS cache poisoning is explained below:
Step-1) First we need to know the expiry time of cached entries(ttl = 0). This can be easily known by using tools like nslookup or dig.

```
root@rk:~# nslookup -type=A -debug geeksforgeeks.org
Server:          192.168.1.1
Address:192.168.1.1#53
------------
   QUESTIONS:
     geeksforgeeks.org, type = A, class = IN
   ANSWERS:
   ->  geeksforgeeks.org
     internet address = 52.25.109.230
     ttl = 24
   AUTHORITY RECORDS:
   ADDITIONAL RECORDS:
------------
Non-authoritative answer:
Name:    geeksforgeeks.org
Address: 52.25.109.230
```

Step-2) The knowledge of time of expiry of cached entries and correct timing to put in the recursive query and responding with counterfeit reply from attacking server is the key to success for DNS poisoning.
Step-3) Once the cached DNS records have expired, recursive queries to authoritative name servers is made by the local server. Attacker grabs this opportunity by flooding forged DNS records. However, the attacking server faces two challenges in order to poison the entries in local name server:
a) The source IP address in the forged packet by attackng server must be same as that of authoritative name server.
b) Name server uses DNS ID to authenticate other name servers. The DNS ID of attacking server must be same as that of authoritative name server.
Solution to above challenges are beyond the scope of paper and cannot be discussed further.
Step-4) Assuming attacking server is able to satisfy both the condition, it successfully poisons the entries of local DNS server and is able to redirect the client to malicious site.

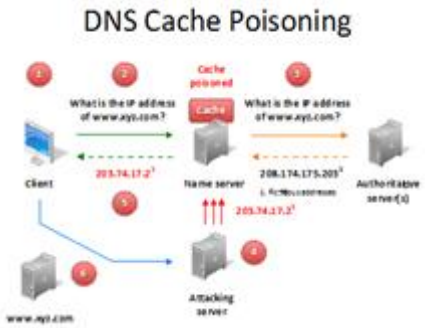

Fig. 5 DNS Cache Poisoning[14]

Till now the attacker has been successful to redirect to malicious site but if the website is using SSL certificate issued by CA the client can detect the attack since SSL certificate authenticates the server to client. However, the use

self-signed SSL certificate makes it very difficult for client whether the server it has been communicating is genuine or not.

Besides these two other attacks there are variety of many other attacks that can be deployed against server using self-signed SSL like replacing javascripts with malicious scritps that can be used to run various exploits on client-side.

V. METHODOLGY FOR MAN-IN-THE-MIDDLE ATTACK ON SELF-SIGNED CERTIFICATE

Setup for the attack:

Attacker Machine: Kali Linux (4.18.0-kali2-amd64 #1 SMP Debian 4.18.10-2kali1 (2018-10-09) x86_64 GNU/Linux).
Network Type: IEEE 802.11 (can be IEEE 802.3 as well).
Self-Signed SSL Certificate used:

Version: 3 (0x2)
Signature Algorithm: sha256WithRSAEncryption
Public Key Algorithm: rsaEncryption

Tools used:
Ettercap 0.8.2
Mitmproxy: 4.0.4
Python: 3.7.2+
OpenSSL: OpenSSL 1.1.1a 20 Nov 2018

Along with these tools we have used some self developed programs for recording, interceptinvg and other purposes. To reproduce the attack complete code can be obtained from git repository: https://github.com/Rishabh0712/NICK.git

The attacker machine connects to the network, redirects all the traffic of network from gateway towards itself(on some port say port X) using ARP poisoning. Then a proxy server running on port X in attacker machine intercepts the traffic and replace the SSL certificates on the fly. This enables the attacker machine to record and read all the SSL encrypted traffic in plaintext.This attack is effective against all the websites that do not use HTTP strict transport security policy.

Workflow to perform man-in-the-middle attack on self-signed SSL:

Published By:
Blue Eyes Intelligence Engineering
& Sciences Publication

VI. RECOMMENDATIONS

1) Using SSL certificate issued by trusted CA can eliminate most of the threats and site administrators must always go for certificate issued by CA instead of using self-signed SSL.
2) Client side attacks over a network lagerly rely on ARP Poisoning for their success. Network Administrators must apply suitable techniques like maintaing static ARP to prevent ARP Poisoning firsthand. Also, detection tools like XArp can be used to detect ARP Poisoning attack.
3) DNS Spoofing is another base attack through which security of sites using self-signed SSL certificate can be compromised. To prevent DNS Spoofing :

a) TTL value must be kept as low as possible.

b) Verification of authoritative name server must be thoroughly done.

c) Name server must not respond to DNS request over WAN on port 53.

4) In all conditions, Intrusion Detection System must be present in the network to detect and prevent attacks on the network.

REFERENCES

1. P.Mohan, J.Anuradha, "Network Security and Types of Attacks in Network", International Conference on Intelligent Computing, Communication & Convergence-2015
2. D.Manik, S.Navkar, "On the security of SSL/TLS-enabled applications", Applied Computing and Informatics,in press.
3. W. Xuetao, W. Michaell, "A Survey of HTTPS Implementation by Android Aps:", 3rd ed., vol. 2. Oxford: Clarendon, 1892, pp.68–73.
4. D.Christian, C.Kim, "A technique to circumvent SSL/TLS validations on iOS devices", Future Generation Computer Systems 74 (2017) 366–374.
5. L.Homin, M.Tal,N.Erich,"Cryptographic Strength of SSL/TLS Servers:Current and Recent Practices"
6. Md, Faisal & Rahman, Abdur & Kamal, Parves. (2014),"A Holistic Approach to ARP Poisoning and Countermeasures by Using Practical Examples and Paradigm",International Journal of Advancements in Technology. 5.
7. Tripathi, Nikhil & Swarnkar, Mayank & Hubballi, Neminath. (2017). "DNS Spoofing in Local Networks Made Easy." 10.1109/ANTS.2017.8384122.
8. Benton, Kevin & Jo, Juyeon & Kim, Yoohwan. (2011). "SignatureCheck: a protocol to detect man-in-the-middle attack in SSL". 10.1145/2179298.2179365.
9. H.Lin-Shung,R.Alex,E.Erling,J.Collin,"Analyzing Forged SSL Certificates in the Wild"
10. https://docs.mitmproxy.org/stable/concepts-howmitmproxyworks/
11. https://www.itec.suny.edu/scsys/vms/ovmsdoc073/v73/6646/the_ssl_protocol.html
12. https://computer.howstuffworks.com/dns3.html
13. http://techgenix.com/understanding-man-in-the-middle-attacks-arp-part2/