Available Online at www.ijcsmc.com

International Journal of Computer Science and Mobile Computing

A Monthly Journal of Computer Science and Information Technology

ISSN 2320–088X

IJCSMC, Vol. 4, Issue. 5, May 2015, pg.566 – 573

RESEARCH ARTICLE

MAN IN MIDDLE ATTACK IN SSL AND HTTPS

Ritesh Kumar Yadav

University School of Information and Communication Technology,

Guru Govind Singh Indraprastha University, NEW DELHI

riteshyadav.cse@gmail.com

ABSTRACT: ***Protecting our data online is never going to be an easy task, especially nowadays when attackers are regularly inventing some new techniques and exploits to steal your data. Sometimes their attacks will not be so harmful for individual users. But large-scale attacks on some popular websites or financial databases, could be highly dangerous. In most cases, the attackers first try to push some malware on to user's machine. Sometimes this technique doesn't work out, however. Most of the effective defenses against MITM can be found only on router or server side. We don't having any dedicated control over the security of our transaction. The ultimate goal of our proposed system is to create secure channels over insecure networks. Secure Socket Layers (SSL), or Transport Layer Security (TLS) in its more modern implementation, is protocols designed to provide security for network communication by means of encryption. This protocol is most commonly associated with other protocols to provide a secure implementation of the service that protocol provides that is HTTPS with secure socket layer.***

Keywords: MIMA, SSL/TLS, Secure Communication

I. INTRODUCTION

With the development of e-commerce and Cloud Computing, SSL protocol is more and more widely used in all kinds of network services. SSL protocol by providing end to end authentication, message encryption, message Integrity check and other security mechanisms protects the security of the communication process. For example, Yahoo ensures the security of the mail account through SSL, to protect the safety of the user e-mail account. Amazon shields the user transaction account and transaction security by it. In recent years, due to the development of cloud computing, the connection security between the client and the cloud is also an extremely important issue. December 2009, VeriSign announced, VeriSign will provide security and authentication services cloud-based computing for

Microsoft Windows Azure platform. And Microsoft will use VeriSign SSL Certificate and VeriSign code signing certificate, to ensure the security of the cloud-based computing services and applications that being developed and deployed on the Windows Azure platform. Because, in the using of cloud computing model, the users` all computing resources are stored in the cloud, so network connection is essential for you to normally work. Therefore, if the server ends were safe enough, the security of network transmission would become very important. The SSL protocol is widely embedded in the client browser currently, the server-side also is relatively easy to deploy and implement, and the SSL protocol itself has good security features. At present, most bases of the cloud security applications are using the SSL protocol, many current hacker communities study SSL, so that the security issues of the SSL protocol are more and more concerned. But SSL is not perfect in the practical application, there still exists the possibility of middle attack. 2003, in the literature[1], Peter Burkholder studied the defects of SSL handshake and verified the possibility of SSL attack, using the way of hijacked session of the typical middle attack to deceive the client to achieve the attack. In the year of 2009, Michael Howard, in the paper[2], showed to carry out SSL attacks by the use of tools based on the Webmitm, which is a tool for SSL attack, and eventually received the data after decryption. The paper[3]shows, in the International security conference in 2009, Moxie brought forward that the HTTPS to the HTTP transfer connections in the practical application would have security problems, causing link substitution attack, but also the client has not traditional security warning signal, so there is a serious hazard. In the literature[4], basing on the analysis of SSL attacks, put forth to improve the way of the browser interaction design to help users find the presence of attack, thereby enhancing safety. In the paper[5], according to the feature about SSL attacks often based on LAN-deceived, proposed ARP deception defense tool designed by their own to enhance LAN security, this is also obtained better results by the comparing experimental. However, there is a problem that this method is not for the SSL protocol itself. In the paper [6] being similar to [4], improved the security of the parties through upgrading the browser interaction interface, In our proposed system we integrate both SSL and HTTPS protocols for avoid man in middle attack

II. Background and Motivation

Multiple browser-based mechanisms have been proposed to detect forged certificates. For instance, browser extensions can keep track of the certificates used by the browser and can detect certificate changes [7, 11]. While simple, the effectiveness of this approach is affected by false positives and lack of user training. A related technique, known as certificate pinning [9], uses a white-list of certificates for important domains that are hardcoded in the browser. This solution is less prone to false positives; however, it is neither flexible nor scalable. A more robust approach is the use of secondary channels such as cellular networks [10] and Tor [8] to obtain additional copies of the server certificate. Assuming that adversaries have no control over the secondary channels, any inconsistency among the certificates received will indicate a possible MITM attack. Unfortunately, this technique has considerable deployment costs and can introduce significant delays to SSL/TLS connection setup.

2.1 The SSL/TLS Protocols and Web Applications

The SSL/TLS protocols [12, 13] are the main security mechanisms used to protect the communications between browsers and web applications. By providing a transparent encryption layer, SSL/TLS guarantee the confidentiality and integrity of the data traveling across the Internet. Moreover, SSL/TLS allow browsers to authenticate web applications servers via X.509 digital certificates [14].

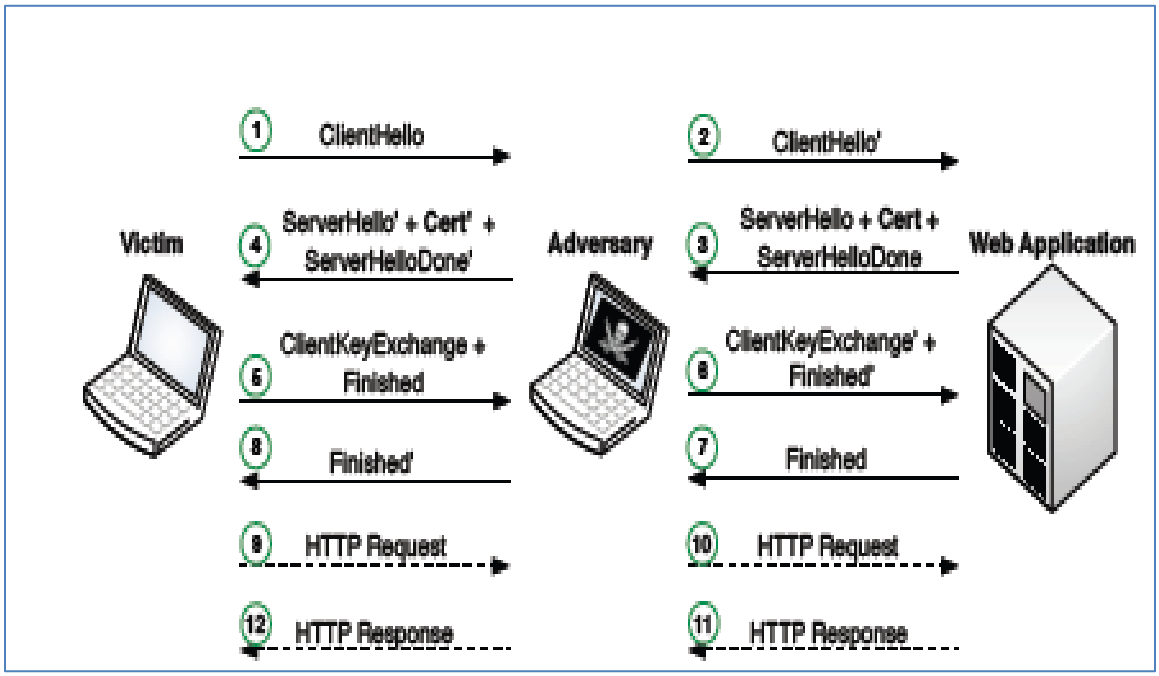

Fig. 1: Example of a MITM attack against SSL/TLS.

A digital certificate binds the server's identity (i.e., domain name) to the server's public key and it is signed by a Certification Authority (CA) trusted by both the server and the browser. CAs are required because the browser and the server do not share any secrets at the SSL/TLS layer; thus, a trusted third-party is needed to vouch for the authenticity of the server's certificate. Certificates can also be used for user authentication; however, this is not a common practice in Internet scenarios. Initially, due to performance considerations, most web applications used SSL/TLS only to protect requests carrying private data (e.g., passwords, credit card numbers). However, due to the increasing number of attacks against web sessions (e.g., session hijacking), many applications have been forced to protect all their communications with SSL/TLS. For this reason, is common that during a session, a browser establishes multiple SSL/TLS connections not only with web application's servers but also with servers from third-party domains (e.g., CDNs and ads networks). Through a short survey from the Alexa Top 20 US sites and popular online banking sites (15 in total), we determined that an average of 12 certificates per domain were validated by the browser, with a minimum of 4 and a maximum of. Moreover, most sites included at least one certificate from a third-party domain.

2.2 MITM Attacks against SSL/TLS

The security guarantees offered by SSL/TLS rely on the correct authentication of the server. All such guarantees are rendered ineffective if an adversary is able to convince users to accept an illegitimately generated certificate, as shown in Figure 1. First, the adversary positions herself in the network path between the victim's computer and the server. When the victim sends a request for establishing a new SSL/TLS connection with the server (message 1), the adversary intercepts and responds to it (message 4) using a forged certificate (Cert'). If the victim accepts this certificate, then she completes the SSL/TLS setup with the adversary (messages 5 and 8), who has, as a result, successfully masqueraded as the server.

Simultaneously, the adversary establishes a new SSL/TLS connection with the server (messages 2, 3, 6, and 7). At this point, the adversary has two active SSL/TLS connections: one with the victim and one with the server.

However, from the victim's and server's perspectives, there is only one secure connection in place. The adversary can now decrypt, re-encrypt and forward all the messages exchanged between the victim and the server (messages 9 to 12). As a result, the adversary can access private information (e.g., passwords) or even modify it (e.g., code injection).

2.3 Problems with Third-Party Solutions

A considerable number of mechanisms have been proposed to improve server-side authentication and protect against MITM attacks. The most popular approach is the use of additional third-party entities that can also vouch for the authenticity of server certificates. Third-party solutions provide a number of benefits: protection of the first connection to a new domain, scalable attestation of certificates for all public domains and minimal requirements for web applications. Unfortunately, this approach also faces several critical challenges. First, these mechanisms have significant deployment and operational costs. The additional infrastructure needed can be expensive to deploy and operate due to requirements such as high-availability, data consistency, performance and security. Even web applications can be affected by the operational overheads required by these mechanisms. Second, the resulting trust model is more complex.

The use of multiple trusted entities to choose from can make the trust model more complex to evaluate and understand. Thus, average users are likely to rely on default trust configurations. Moreover, trust is dynamic a trusted entity today may become an adversary tomorrow. Third, these mechanisms introduce new privacy risks. Users' browsing activity is disclosed to third-party entities. Preventing this problem can add complexity to these solutions. Fourth, certificate revocation procedures become more complex. The use of multiple entities make revocation more difficult because of the additional overhead required to revoke multiple proofs of authenticity (e.g., signatures). Finally, captive portals typically interfere with these mechanisms. In places such as airports and hotels, captive portals can block requests for certificate validation to external entities before user registration. Thus, captive portals need to be modified to allow additional certificate validation mechanisms.

III. SSL and HTTPS

Secure Socket Layers (SSL), or Transport Layer Security (TLS) in its more modern implementation, are protocols designed to provide security for network communication by means of encryption. This protocol is most commonly associated with other protocols to provide a secure implementation of the service that protocol provides. Examples of this include SMTPS, IMAPS, and most commonly HTTPS. The ultimate goal is to create secure channels over insecure networks.

In this paper we will focus on attacking SSL over HTTP, known as HTTPS, because it is the most common use of SSL. We may not realize it but you probably use HTTPS daily. Most popular email services and online banking applications rely on HTTPS to ensure that communications between your web browser and their servers in encrypted. If it weren't for this technology then anybody with a packet sniffer on your network could intercept usernames, passwords, and anything else that would normally be hidden.

The process used by HTTPS to ensure data is secure centers around the distribution of certificates between the server, the client, and a trusted third party. As an example let's say that a user is trying to connect to a Gmail email account. This involves a few distinct steps, which are briefly simplified in Figure 2

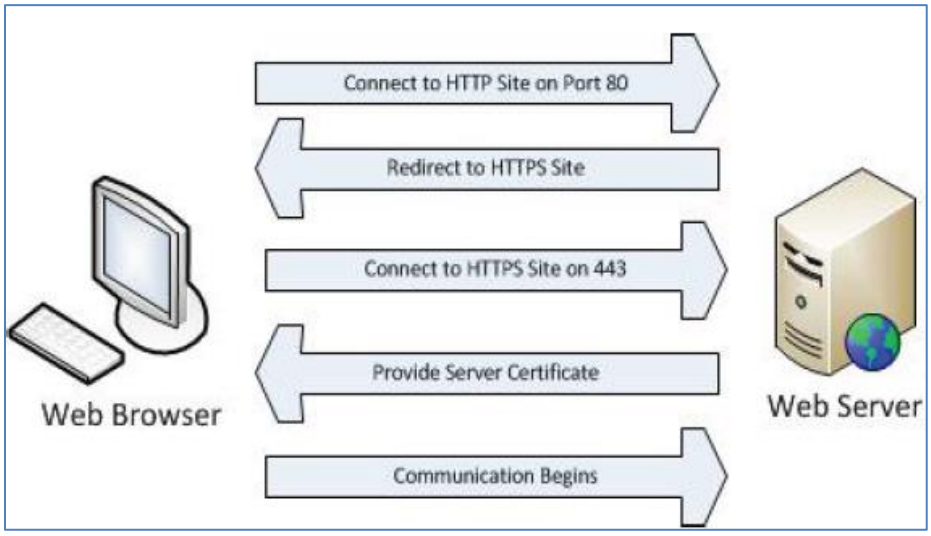

Fig 2: HTTPS Communication Process

Defeating HTTPS

This process was considered highly secure up until several years ago when an attack was published that allowed for successful hijacking of the communication process. This process doesn't involve defeating SSL itself, but rather, defeating the "bridge" between nonencrypted
and encrypted communications. Moxie Marlinspike, a well-known security researcher hypothesized that in most cases, SSL is never encountered directly. That is, most of the time an SSL connection is initiated through HTTPS it is because someone was redirected to an HTTPS via an HTTP 302 response code or they click on a link that directs them to an HTTPS site, such as a login button.
The idea is that if you attack the transition from an unsecured connection to a secure one, in this case from HTTP to HTTPS, you are attacking the bridge and can man in the middle an SSL connection before it even occurs. In order to do this effectively. The process is fairly straightforward and is reminiscent of some of the attacks. It is outlined in Figure 3

The process outlined in Figure 3 works like this:

1. Traffic between the client and web server is intercepted.
2. When an HTTPS URL is encountered sslstrip replaces it with an HTTP link and keeps a mapping of the changes.
3. The attacking machine supplies certificates to the web server and impersonates the client.
4. Traffic is received back from the secure website and provided back to the client.

The process works quite well and as far as the server is concerned it is still receiving the SSL traffic it wants to, it doesn't know the difference. The only visible difference in the user experience is that the traffic will not be flagged as HTTPS in the browser, so a cognizant user will be able to notice that something is amiss.

IV. Analysis and comparison about three methods (SSL,HTTPS,Integrated)

1) The preparation for attacks

The similarity between the first and second method is both of them must use the ARP Spoofing. The third method does not depend on it. With the development of Antivirus Software, many of them can detect the ARP Spoofing, and even some can deal with it in special method, except the third method.

2) Protocol with each side

In first method, the attacker establishes both SSL connections with the client and the server. In the second method, the attacker establishes HTTP connection with the server but HTTP connection with the client. But in the third method, it doesn't establish connection with each side. The device just modifies the certificate and the handshake message for the following communication. The connection after attack is the same to the normal connection. The device just forwards the SSL encrypted data, if needing, it can decrypt the encrypted data.

3) Analysis of attack

The first and third method monitors the HTTPS request, after tampering the certificate, there is a dialog in the browser. If users accept the fake certificate, all the following are the same as the normal connection. On the contrary, the second one is different. It monitors the HTTP streams, and tampers redirection of https address. After Attack, the browser of the client will have no alert, because it is actually a HTTP connection. Though it seems perfect, it has two distinct differences with the normal connection. One is the head of URL, and the other is the lock icon on the browser. If users are aware, the attack fails.

4) Harmfulness

In the first and third method, the success depends on the vigilance of the users. Only when the user clicks on the yes button, the attack can be successful. But in the second method, because of no alert dialog, users can't discern easily. Especially, a few of E-mail system only conduct HTTPS connection to send ID and password, after verifying them quickly redirect to the HTTP connection.

In this case, it seems hardly note the existence of attack. In the test experiment, 30 people through the client computer verify the harmfulness. The statistic data is shown in the TABLE I below:

Table1: RESULT OF ATTACK

Result	*SUCESSFUL*	*FAILED*
SSL	21	9
HTTP	20	10
SSL+HTTPS	18	12

Experimental results show that the number of the success of the first and second method is more than the third method, Hence third is less harmful.

5) Difficulty

Method 1 and Method 2 can be completed by Programming in the local machine, so the implementation does not require attached factors. But method 3 will need the programs to be run on the development board, the runtime library need to be migrated to the dotnet platform, and it's relatively more complicated to implement.

However, given the host owning the firewall with the ARP client protection function, so the third method will be completely free to consider the impact of such factors, but once the ARP deception fails, there is not only the warning of the ARP attack in the client, but also the user may sense to attack there. Then the software will prompt the user to take appropriate measures, such as prompting the user to use a static ARP table, and sometimes the first and second methods may also not succeed.

In the test experiment, 30 people use the client Computer which has been installed special firewall that can defeat ARP Spoofing. The statistic data is shown in the TABLE II below.

Table 2.RESULT OF ATTACK

RESULT	SUCESSFUL	FAILED
SSL	15	15
HTTP	18	12
SSL+HTTPS	22	8

Experimental results show that the success of the third method is more than the first and second method. Because the first and second method depend both on the ARP spoofing, so their results seem similar in this test.

V. Conclusion

Experiments show that three protocol SSL, HTTP and hybridization of ssl and https of attack on the session are feasible. In normal, connection speed of HTTPS services is 2-100 times slower than normal HTTP connection, users will not be aware of attacks even if the delay caused by the change of link. Because users usually don't care about the alert in the browser, when attacking in the first and third method, the majority of users will tend to accept a warning certificate even if the alert dialog; when attacking in second method, the user will not be aware because the little difference between the normal and attacking pages. In our experiment, using SSLHTTPS effectively to avoid the attack. Configuring a static ARP table can avoid attack in first and second method how prevent man-in-the-middle attacks on HTTPS session more effective is the next focus of our study.

REFERENCES

[1] Peter Burkholder, "SSL Man-in-the-Middle Attacks", SANS Institute InfoSec Reading, 2003.

[2] Michael Howard, "Man-in-the-Middle Attack to the HTTPS Protocol", IEEE computer society, 2009, pp.78- 81.

[3] MarlingspikeMoixe, "New Tricks For Defeating SSL in Practice", BlackHatConference, USA(2009).

[4] Haidong Xia and Jose Carlos Brustolonl, "Hardening Web Browsers Against Man-in-the-Middle and Eavesdropping Attacks". Proc. 14th Int'l Conf. World Wide Web(IW3C2), ACM Press, 2005, pp. 489-498.

[5] ThawatchaiChomsiri, "HTTPS Hacking Protection", 21st International Conference on Adcanced Information Networking and Applications Workshops(AINAW), 2007.

[6] Andre Adelsbach and ebatianGajek, "Visual Spoofingingof SSL Protected Web Sites and Effective Countermeasures", Proceedings of the 1st Information Security Practice and Experience Conference, Singapore, 11-14 April, 2005.

[7].Certificate Patrol (2010), http://patrol.psyced.org/

[8]. Alicherry, M., Keromytis, A.D.: DoubleCheck: Multi-path Verification Against Man-in-the- Middle Attacks. In: Proceedings of the IEEE Symposium on Computers and Communications (2009)

[9]. Evans, C., Palmer, C.: Certificate Pinning Extension for HSTS (2011), http://www.ietf.org/mailarchive/web/websec/current/pdfnSTRd9kYcY.pdf

[10]. Parno, B., Kuo, C., Perrig, A.: Phoolproof Phishing Prevention. In: Proceedings of FinancialCryptography and Data Security (2006)

[11]. Soghoian, C., Stamm, S.: Certified Lies: Detecting and Defeating Government Interception Attacks Against SSL. In: Proceedings of Financial Cryptography and Data Security (2011)

[12]. Dierks, T., Rescorla, E.: RFC 5246 - The Transport Layer Security (TLS) Protocol Version 1.2 (2008), http://tools.ietf.org/html/rfc5246

[13]. Freier, A., Karlton, P., Kocher, P.: RFC 6101 - The Secure Sockets Layer (SSL) Protocol Version 3.0 (2011),

[14]. Adams, C., Farrell, S.: RFC 2510 - Internet X.509 Public Key Infrastructure Certificate Management Protocols (1999)

