Journal of Internet Banking and Commerce

An open access Internet journal (http://www.icommercecentral.com)

Journal of Internet Banking and Commerce, August 2017, vol. 22, no. 2

TLS PROTOCOL VERIFICATION FOR SECURING E-COMMERCE WEBSITES

DAASSA ASMA

Electronics and Microelectronics Laboratory, Faculty of Sciences, Monastir, National Engineering School of Tunis, University of Tunis El Manar, Tunisia

Tel: +21692017999;

Email: **asma.daassa@gmail.com**

MACHHOUT MOHSEN

Electronics and Microelectronics Laboratory, Faculty of Sciences, University of Monastir, Monastir, Tunisia

AGUILI TAOUFIK

Syscom Laboratory, Department of Information and Communications Technology, National Engineering School of Tunis, Tunis, Tunisia

Abstract

E-commerce security is very important especially nowadays but internet is entrusted due to the attacks and hackers exploitations. To improve the security of electronics transactions, many protocols are developed. SSL/TLS is the most commonly used, although many dangerous attacks were found. So, developers have to upgrade

SSL/TLS to avoid these attacks and enhance security.

To achieve their goals, hackers exploit flaws and errors found in SSL/TLS protocol implementations, it is necessary to verify and validate the security of the entire software code. Therefore, to improve the security of SSL/TLS protocol, researchers try to find solutions; protocols must therefore be tested and validated before their launch. In this paper, we will focus on analyzing SSL/TLS protocol with automated formal verification tool AVISPA. We study the renegotiation attack and try to detect it using AVISPA. We use formal models for automatic verification of security protocol to discover new attacks, to prevent similar attack in the future and also to increase the tool efficiency.

Keywords: **SSL/TLS; Security; e-commerce; HLPSL; AVISPA; Attacks**

INTRODUCTION

Strengthening the security of e-commerce transactions is indispensable to prevent advanced attack techniques. Data exchange during an electronic transaction may be unsafe due to attacks and hacker's exploitations.
The spread of internet services for professionals and individuals: e-commerce, bank account management, sensitive data and personal information (credit card number, password) etc., make providing and improving the security of electronic transactions very important and necessary.
To ensure the confidentiality and integrity of these electronic transactions, many security protocols have been developed, but it is difficult to ensure its rigorous analysis and validation. As a result, hackers exploit these flaws and vulnerabilities to gain access to sensitive data and information.
In order to reduce these risks, an analyzing tool for cryptographic protocols is developed "AVISPA" which stands for Automated Validation of Internet Security Protocols and Applications.

To detect flaws and vulnerabilities in security protocols specifications, many techniques are used like penetration test and fuzz test.

Contributions

The contributions of this paper can be summarized as follows: -We focus on the validation of SSL/TLS implementations, and we use the AVISPA tool for automated verification and analyzing security properties.
1. We study the renegotiation attack and triple handshake. Our purpose is to detect the renegotiation attack using AVISPA.
2. We discuss countermeasures.

Therefore, in this work, we analyze TLS handshake using AVISPA, we try to compare the different tool's back-ends and using mutation techniques we try to detect the renegotiation attack.

Initially, we start from a secure formal modal of SSL/TLS protocol. The first phase of our work is to do some modification in the HLPSL model of our protocol, we try to find attack traces provided by different back-ends and we compare its different results.

The second phase is not an original contribution from ours but we choose a protocol commonly used especially nowadays in electronic transactions, following [15], [16], [21], we apply several mutation techniques into the HLPSL modal to create vulnerabilities, then we use model checking to verify the protocol and generate attack traces.

Paper organization

The remainder of the paper is organized as follows. Section 2 presents e-commerce security issues and an overview of SSL/TLS protocol and also its recent attacks, section 3 reviews related work, discusses the validation of security protocols, section 4 introduces the formal analysis of SSL/TLS protocol, and section 5 presents formal analysis for attack detection . Finally, we conclude the paper in section 6.

BACKGROUND

There are many e-commerce security protocols, SSL/TLS, SET, 3D secure (...). For example, SET suffers from many problems making it difficult to be used in the real world. In fact, this protocol needs a complicated infrastructure and it is a complex protocol. That's why, we focus our study on SSL/TLS protocol which is the most popular and used protocol for securing e-commerce transactions.

To encourage customers making goods and items purchase, we have to improve the security of electronic transactions, that's why researchers study in depth different attacks and propose solutions to these problems.

In this section we introduce some background information about SSL/TLS protocol for securing communications and electronic transactions, e-commerce security issues and a detailed chronology of recent attacks, renegotiation attack, Triple handshake attack.

Overview of the SSL/TLS protocol

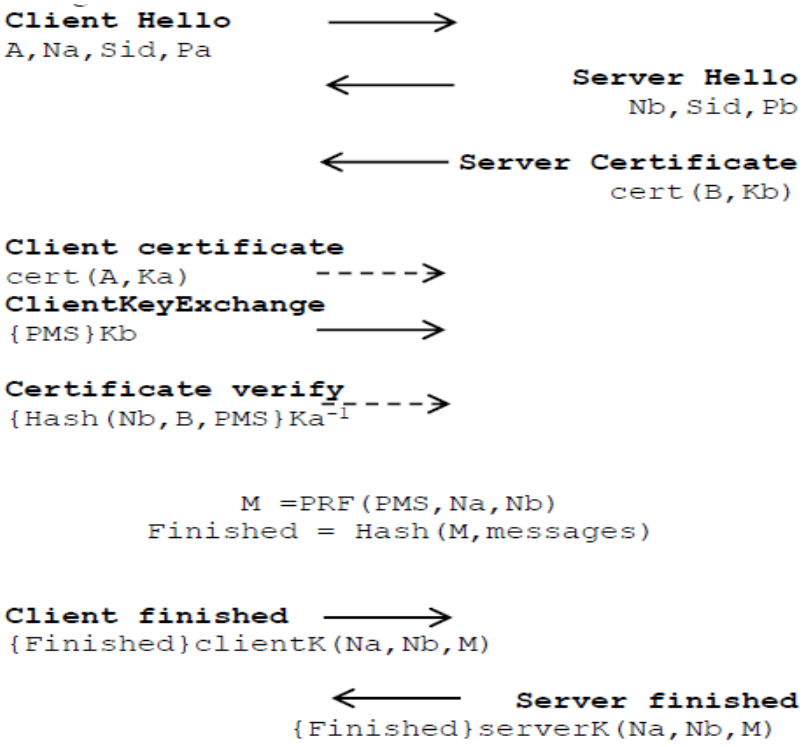
SSL (Secure Socket Layer) is a security protocol used to provide secure communications between client and server, so all data exchanged between them are private and integral. Many websites use this protocol to ensure protection of their

online transactions.

We illustrate the SSL/TLS handshake and we present different messages exchanged between a client and a server

SSL/TLS has two layers: the lower layer is the record protocol, it allows exchanging messages using current connection settings, and the upper layer is the handshake protocol. It allows authentication of client and server with certificate-based server authentication and optionally client authentication, the negotiation of encryption and hash algorithms and a shared master secret. Both client and server agree on their exchanged settings (Figure 1).

Figure 1: TLS handshake.



E-commerce security issues and recent attacks

Although, the number of attacks is still growing, SSL/TLS is the most used protocol in e-commerce.

To enhance electronic transactions security, a lot of studies analyze a number of vulnerabilities and attacks on SSL/TLS recently discovered [1-3]. Among these vulnerabilities we can list BEAST, Lucky13, CRIME, TIME and BREACH, [1] and [4] studied and discussed the MITM attack. We can say that possibly Heart bleed is the most dangerous attack on SSL/TLS protocol, the attacker exploits this bug to get

sensitive data and personal information without leaving a trace [5-8]. According to Qualys Labs' SSL pulse survey published in December 03, 2015 , 326 sites are still vulnerable to the heart bleed bug and 90,4% sites tested are vulnerable to the BEAST attack [9].

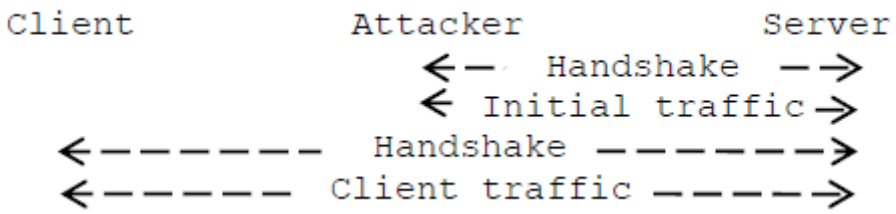
In 2015, many attacks were discovered on SSL/TLS protocol: the logjam attack [10], the FREAK attack [11], like Heart bleed, hackers could steal many critical information. Also, using perfect forward secrecy PFS was not efficient to secure old traffic due to the discovery of new attack called the Logjam Attack. In [12], they discovered another flaw in TLS protocol which allows MITM attack, and they proved that Diffie Hellman is not really secure. In fact, they broke the 512-bit Diffie Hellman, so they named it imperfect forward secrecy. Throughout recent years, much vulnerability in SSL/TLS has been revealed. Therefore, the e-commerce security became very exhausting especially due to many flaws found in security protocol implementation.

The importance of validating the security of these implementations serves to prevent flaws and attacks exploited by hackers and improve e-commerce security. We choose to study in depth the SSL/TLS renegotiation attack. And we also focus on triple handshake attack which is discovered recently in 2014.

Renegotiation attack (CVE-2009-3555)

Using AVISPA tool, we try to detect the TLS renegotiation vulnerability (RFC 5246) [3] which is a flaw in the protocol. This option has complicated the protocol, but it is important to get authentication of both client and server. This flaw is due to the problem of binding between the previous session (before renegotiation) and the new one. As a solution to this attack (RFC 5746), researchers proposed creating a connection between the old and new session by adding an extension called renegotiation info. But this solution is not evident because old servers do not know this extension (Figure 2).

Figure 2: Renegotiation attack.



Triple handshake attack (CVE-2014-1295)

This is a serious flaw discovered recently in the TLS protocol that offers the

possibility to attackers to intercept and decrypt communications. Mounting MITM attack on the protocol is feasible Using triple handshake attack.

Moreover, a successful attack can lead to MITM and break secure renegotiation. Between the honest client and server, malicious servers exploit man-in-the-middle attack on three successive handshakes, and impersonate the client on the third handshake.

In this attack, two TLS sessions use the same key, allowing several other attacks. Renegotiation attack exploits the lack of binding between two handshakes on the same connection, triple handshake exploit a problem of binding when TLS sessions are resumed on new connections.

So, RFC5746 cannot be a solution for triple handshake to prevent this attack. This work [13] proposed binding the master secret and the session resumption handshake to the full handshake.

RELATED WORK

There are several works aimed to ensure the robustness and the validity of security protocols.

Recently, some researchers chose to use formal verification of security protocols, but this method is not enough to improve the security of protocols implementations. To achieve their goal, many researches study in depth the validation of security protocols using many verification techniques [14-16].

The SSL/TLS protocol suffers from several types of attacks, like error implementations and cryptographic flaws. Due to the importance of this protocol and its use in several areas, especially e-commerce, several studies have been implemented to improve the security implementations.

For example, the most famous attack called Heart bleed spread quickly and put users' passwords at many popular Web sites at risk, because many tools and techniques for detecting security flaws didn't succeed. Heart bleed is a buffer over-read vulnerability, but it is not a buffer over-writes. That's why, for example fuzzers, traditionally applied to find buffer over-writes, failed to find it. So Heart bleed created a challenge for tools to discover it [17].

Several methods were used for the verification of cryptographic protocols, but verification of these security protocols, even if it is feasible manually on small protocols, became difficult to apply on complex protocols like SSL/TLS and SET. That's why we choose to use automatic formal verification to ensure some security properties like authentication and secrecy properties.

Some works chose to apply the verification using automated tools dedicated to this effect such as ProVerif [19], Scyther [20] AVISPA [21], while, others [14,21] verified the implementation of the protocol. For example, [19] uses VCC to prove protocol security and validate its implementation in C language.

Using AVISPA tool, much work has been performed for the verification of cryptographic protocols based on the abstraction technique. Indeed, this technique is to make abstractions of the implementation details of the protocols and focus on the mechanisms that establish the security properties to consolidate the formal specification.

Once the specification is written, it can be analyzed using the AVISPA tool to discover vulnerabilities if they exist. The tool generates attack traces invalidating the declared properties and vulnerability is confirmed in the specification.[14],[15],[20].

Some works are based on a very special technique, which appeared recently, but has potential and a promising future. This process is known as mutation testing [15], [17], [21]. This technique involves injecting faults in a HLPSL model, which uncovers vulnerabilities, and then trying to detect them. These mutations can simulate errors caused by a programmer.

Therefore, injecting faults into programs or models is a technique called mutation testing. It is used for many purposes such as evaluating the quality of existing software tests and improving its efficiency in discovering defects [22,23].

Analysis of the TLS Handshake Protocol with AVISPA

We will analyze TLS Protocol used to establish a secure connection between client and server. TLS provides authentication and confidentiality between two agents. TLS handshake serves to negotiate encryption algorithms, secret symmetric keys, and to authenticate agents to each other.

The formal model of SSL/TLS protocol is the following:

1: A -> B: A, Na, Sid, Pa
2: B -> A: Nb, Sid, Pb
3: B -> A: {B, Kb}inv(Ks)
4: A -> B: {A, Ka}inv(Ks)
5: A -> B: {PMS}Kb
6: A -> B: {H(Nb,B,PMS)}inv(Ka)
7: A -> B: {Finished}Keygen(A, Na, Nb, M)
8: B -> A: {Finished}Keygen(B, Na, Nb, M)

This model shows us messages exchanged between agents in SSL/TLS protocol.

We will now explain some steps.

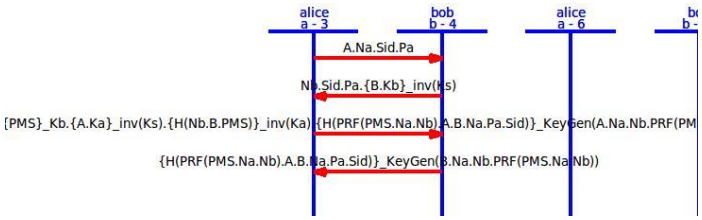
Step 1: in this step, A generates fresh nonce Na. A determines Sid which is necessary for session resuming; we assume that Sid is randomly generated by A, and A chooses options Pa

Choosing encryption options Pa and Pb (0 in steps 1 and 2) are used in the TLS specification to establish the preferences for encryption and compression. we assume that Pa and Pb are just some randomly generated data.

SPAN (Security Protocol Animator for AVISPA) is used to animate HLPSL specifications, producing Message Sequence.

Charts (MSC) which can be seen as an “Alice and Bob” trace of an HLPSL specification (Figure 3).

Figure 3: Basic animation of TLS handshakespecification.



Since our work focuses on SSL/TLS protocol, we modified the HLPSL model of TLS protocol taken from AVISPA library [12].

We use AVISPA tool to analyze TLS handshake.

The goals that should be achieved by the protocol are secrecy of symmetric client key, secrecy of symmetric server key, Alice authenticates Bob and Bob authenticates Alice.

Using the keywords (secret, witness, request, and request) in the transition state, the section goal defines the security properties such as mutual authentication and established secret key between the client and the server. Therefore, the available keywords are used to verify the strong or weak authentication of agents and the secrecy of some information.

goal

secrecy_of sec_clientk,sec_serverk %Alice authenticates Bob on na_nb1
authentication_on na_nb1
%Bob authenticates Alice on na_nb2 authentication_on na_nb2

end goal

In order to specify the authentication goal through AVISPA tool, we exploit witness and request command. And to specify the secrecy goal, we make use of the secret command. Now we are going to discuss these commands.
The authentication between agents can be attained using these statements.

/
witness(A,B,na_nb2,N
a.Nb') [1a]
/
witness(B,A,na_nb1,N
a'.Nb') [1b]

The meaning of the statement [1a] is as the following:
1. Agent A creates Na.Nb' value for agent B
2. Agent A wants agent B to agree to the value.
To be secure from intruder, the value created is for the na_nb2 purpose

In the same way, the meaning of the statement [1b] is as the following:
1. Agent B creates Na.Nb' value for agent A
2. Agent B wants agent A to agree to the value.
To be secure from intruder, the value created is for the na_nb1 purpose.

/\ request(A,B,na_nb1,Na.Nb)[2a]
/\ request(B,A,na_nb2,Na.Nb)[2b]

The meaning of the statement [2a] is as the following :
1. Agent A believes that he communicates with agent A
2. Agent A accepts the value of *Na_Nb* for the *na_n1* purpose

In the same way, the meaning of the statement [2b] is as the following :
1. Agent B believes that he communicates with agent A
2. Agent B accepts the value of *Na_Nb* for the *na_n2* purpose

The secrecy goal can be attained using these statements using the keyword secret.

/\ secret(ClientK,sec_clientk,{A,B})
/\ secret(ServerK,sec_serverk,{A,B})

The symmetric client key and the symmetric server key remain secret between Alice and Bob

Through Avispa, we analyzed TLS handshake and we tried to compare results of four back-ends OFMC (On−the−fly Model−Checker), CLAtSe(CL−based Attack Searcher), SATMC.

(SAT−based Model−Checker) and TA4SP (Tree Automata based Protocol Analyzer).

We modified the HLPSL model to have a TLS handshake without certificate check and with one symmetric key, and we analyzed through AVISPA using different back-ends, then we tried to note the result of each one.

The formal model of TLS handshake without certificate check is the following:

A -> B: A, Na, Sid, Pa
B -> A: Nb, Sid, Pb B -> A: B, Kb
A -> B: {A, Ka}inv(Ks)
A -> B: {PMS}Kb
A -> B: {H(Nb,B,PMS)}inv(Ka)
A -> B: {Finished}Keygen(A, Na, Nb, M)
B -> A: {Finished}Keygen(B, Na, Nb, M)

The corresponding attack model is the following:

A -> I: A, Na, Sid, Pa
I -> A: Ni, Sid, Pi
I -> A: B, Ki
A -> I: {A, Ka}inv(Ks)
A -> I: {PMS}Ki
A -> I: {H(Ni,B,PMS)}inv(Ka)
A -> I: {Finished}Keygen(A, Na, Nb, M)
I -> A: {Finished}Keygen(B,Na, Nb, M)

The formal model of TLS handshake with one symmetric key is the following:

A -> B: A, Na, Sid, Pa
B -> A: Nb, Sid, Pb
B -> A: {B, Kb}inv(Ks) A -> B: {A, Ka}inv(Ks) A -> B: {PMS}Kb
A -> B: {H(Nb,B,PMS)}inv(Ka)
A -> B: {Finished}Keygen(Na, Nb, M)
B -> A: {Finished}Keygen(Na, Nb, M)

The corresponding attack model is the following:
A -> I: A, Na, Sid, Pa
I ->B: A, Ni, Sid, Pi
B -> I: Nb, Sid, Pi
B -> I: {B, Kb}inv(Ks) I -> A: Ni, Sid, Pa
I -> A: {B, Kb}inv(Ks)
A -> I: {A,Ka}inv(Ks)
A -> I: {PMS}Kb
A -> I: {H(Ni,B,PMS)}inv(Ka)

A -> I: {Finished}Keygen(Na, Nb, M)
I -> A: {Finished}Keygen(Na, Nb, M)

Table 1: Results of four back-ends.

Back-end	**One symmetric key**	**Without certificate check**
OFMC	Unsafe	Unsafe
CLAtSe	Unsafe	Unsafe
SATMC	Safe	Safe
TA4SP	Inconclusive	Inconclusive

Finally, we notice from these results that SATMC and TA4SP are useless. SATMC always returns SAFE and TA4SP always returns INCONCLUSIVE, the two back-ends don't provide trace and don't succeed to find attacks. That's why we conclude that results provided by these back-ends are extremely useless (Table 1).

However, OFMC and CLAtSe found attacks and also provided trace. We notice that OFMC is very fast and very reliable, and CL-AtSe is just as reliable, but slightly slower.

Therefore, we conclude that results provided by these back-ends are extremely useful, and chose to use OFMC in our work.

FORMAL ANALYSIS FOR ATTACK DETECTION

To prove our security protocol, we use the Automated Validation of Internet Security Protocols and Applications (AVISPA).

As specification language, AVISPA uses HLPSL (High Level Protocol Specification Language) that is used to specify our security protocol in roles.

The HLPSL is a role-based language, a role describing actions of each agent. There are many types of roles: agent, session, and environment.

HLPSL is used also to describe the initial knowledge of an intruder and to specify the protocol's security properties.

Then we use HLPSL language for specification description. Our goal is to detect the SSL/TLS renegotiation vulnerability,

We use the AVISPA platform to analyze the model in HLPSL and verify if the security goals are satisfied or not using its different back-ends.

These back-ends generate the attack trace, if a specification goal is violated.

Our goal is to generate model-based test to validate the implementation of this protocol.

To describe the security goals such as secrecy or authentication in HPSL, we built the role goal and for authentication, we use two terms which are witness and request.

We use AVISPA tool to verify TLS protocol, and we try to make sure of the detection of the renegotiation vulnerability in the result verification.

We built a formal model to allow detection of the renegotiation attack, which has two roles (the client and the server). During the verification, an agent plays the given role and acts according to the assumed role definition. We use the name Alice to denote the client role, while the name of the agent playing the role will be called A. We use the name Bob to denote the server role, while the name of the agent playing the role will be called B. The agent A tries to do the initial negotiation and renegotiation with the agent B. The server role provides the functionality to allow Client negotiation and renegotiation.

Our next step is to add the fix to the TLS renegotiation problem into TLS AVISPA model and verify that the vulnerability is resolved. The fix is to add a TLS renegotiation extension. Another solution to fix this problem is to make the client authentication not optional, and verify that their identities are still the same after each renegotiation.

Therefore, in our work, we follow the following steps:

1. Formal specification of SSL/TLS protocol using HLPSL language.
2. We apply mutations into HLPSL model using an existing mutant generator jMuHLPSL[18], it consists of injecting logical faults in order to create vulnerabilities.
3. Then we use AVISPA tool and CL-AtSe model-checker to generate attack trace if the mutant is declared unsafe.

Attacker Model and Attack trace

The formal cryptographic analyzers implement Dolev-Yao intruder model.

With Dolev-Yao model, an intruder:
1. has a complete control of the network
2. can intercept all messages sent by the various participants
3. is also able to edit messages and send data by usurping the identity of other legitimate or honest participant.

We model a successful run of the protocol and verify the secrecy and authentication using OFMC and CL-AtSe back-ends and we try to compare their results.

The attack trace of the renegotiation vulnerability should be as the following:

I->B: start
I->B: (I.Ni.Sid.Pi1)
B->I: (Nb.Sid.Pb)

I->B: pair{crypt(Kb,PMS),crypt(IntruderK,Finished)}
B->I: crypt(IntruderK,Finished1)
I->B: crypt((I.Ni.Sid.Pi2), IntruderK)

DISCUSSION

Self-signed and testing

We now discuss some recommendations to e-commerce websites developers to avoid the vulnerabilities discussed in this paper.

Avispa is useful to find attacks and to validate security protocols, but this tool is limited to specify the authentication goal using witness and request command, and the secrecy goal using secret command. However, e-commerce security protocols have to verify more security properties such as non-replay, non-repudiation...

Specifying more security properties is still an open issue and an important main line of research.

Although the Automated Validation of security protocols tools made improvements to specify more security properties, they are still very useful in practice to detect very early vulnerabilities which would be impossible to correct in later phases.

Open problems

Following our analysis, we are trying to highlight few open research problems. First we will discuss the need to provide developers with more effective tools that can help them detect and fix issues before using SSL/TLS protocol in order to secure e-commerce websites. Second we will talk about the ability to specify more security properties through AVISPA.

CONCLUSION

The number of vulnerabilities on SSL/TLS protocol is growing, especially flaws and errors found in existing implementations. Trying to validate these implementations is very important, that’s why we are using the AVISPA automated verification tool, in this paper.

For our future work, we plan to study in depth and to expand our analysis to other e-commerce security protocols like SET.

REFERENCES

1. Pathak M, Raza N (2014) Comprehensive Analysis of Man in the Middle Attack and Propose Statistical Detection Approach. International Journal of Advanced Research in Computer Science and Electronics Engineering (IJARCSEE), p: 269.
2. Sarkar PG, Fitzgerald S (2013) Attacks on SSL a comprehensive study of beast, crime, time, breach, lucky 13 and rc4 biases. ISEC Partners.
3. Meyer C, Schwenk J (2013) Lessons learned from previous SSL/TLS attacks-a brief chronology of attacks and weaknesses. IACR Cryptology ePrint Archive.
4. Das ML, Samdaria N (2014) On the security of SSL/TLS-enabled applications. Applied Computing and Informatics 10: 68-81.
5. Gujrathi S (2014) Heart bleed bug: Anopenssl heartbeat vulnerability. International Journal of Computer Science and Engineer Science and Engineering 2: 61-64.
6. Tsoutsos NG, Maniatakos M (2014) Trust No One: Thwarting" heartbleed" Attacks Using Privacy-Preserving Computation. In VLSI (ISVLSI), 2014 IEEE Computer Society Annual Symposium on (pp. 59-64). IEEE.
7. Mpofu TP, Elisa N, Gati N (2014) The Heartbleed Bug: An Open Secure Sockets Layer Vulnerability.
8. Momani EMH, Hudaib AAZ (2014) Comparative Analysis of Open-SSL Vulnerabilities and Heartbleed Exploit Detection. International Journal of Computer Science and Security (IJCSS) 8: p. 159.
9. https://www.trustworthyinternet.org/ssl-pulse/
10. https://weakdh.org/
11. https://freakattack.com/
12. http://www.avispa-project.org
13. Maatoug G, Dadeau F, Rusinowitch M (2014) Model-Based Vulnerability Testing of Payment Protocol Implementations. HotSpot'14-2nd Workshop on Hot Issues in Security Principles and Trust, affiliated with ETAPS 2014.
14. Adrian D, Bhargavan K, Durumeric Z, Gaudry P, Green M, et al. (2015) Imperfect forward secrecy: How Diffie-Hellman fails in practice. Proceedings of the 22nd ACM SIGSAC Conference on Computer and Communications Security, pp: 5-17.
15. Bhargavan K, Fournet C, Corin R, Zalinescu E (2008) Cryptographically verified implementations for TLS. Proceedings of the 15th ACM conference on Computer and communications security, ACM, pp: 459-468.
16. Bhargavan K, Lavaud AD, Fournet C, Pironti A, Strub PY (2014) Triple handshakes and cookie cutters: Breaking and fixing authentication over TLS. Security and Privacy (SP), 2014 IEEE Symposium-IEEE, pp: 98-113.
17. Maatoug G, Dadeau F, Rusinowitch M (2014) Model-based vulnerability

testing of payment protocol implementations. HotSpot'14-2nd Workshop on Hot Issues in Security Principles and Trust, affiliated with ETAPS 2014.
18. Dadeau F, Héam PC, Kheddam R (2011) Mutation-based test generation from security protocols in HLPSL. Software Testing, Verification and Validation (ICST), 2011 IEEE Fourth International Conference-IEEE, pp: 240-248.
19. Blanchet B (2009) Automatic verification of correspondences for security protocols. Journal of Computer Security 17: 363-434.
20. The Scyther Tool: Verification, falsification, and analysis of security protocols.
21. Viganò L (2006) Automated security protocol analysis with the AVISPA tool. Electronic Notes in Theoretical Computer Science 155: 61-86.
22. Polikarpova N, Moskal M (2012) Verifying implementations of security protocols by refinement. Verified Software: Theories, Tools, Experiments, Springer Berlin Heidelberg, pp: 50-65.
23. Kupsch JA, Miller BP (2014) Why do software assurance tools have problems finding bugs like heartbleed?. Continuous Software Assurance Marketplace.