International Journal of Modern Engineering Research (IJMER)
www.ijmer.com Vol. 3, Issue. 5, Sep - Oct. 2013 pp-3096-3106 ISSN: 2249-6645

# Network Forensic Investigation of HTTPS Protocol

Manesh T[1], Brijith B[1], Bhraguram T M[1], R Rajaram[1], Bhadran V K[2]
[1]*Shankara Centre for Research in Information Science*
*Adi Shankara Institute of Engineering & Technology,Kalady, Ernakulum, Kerala-683574,*
[2]*Resource Center for Cyber Forensics CDAC Trivandrum, Kerala-695033*

***ABSTRACT :*** *Nowadays a large amount of personal and business transactions are done electronically through secured internet communication with HTTPS Protocol. The internet offers computer users access to a wealth of information and reaches in to the heart of many organizations. In this context, there are many possibilities for having different malicious activities or attacks that may occur through the HTTPS protocol. Usually it is very difficult to see or recreate HTTPS network sessions to verify its content as part of the forensic analysis. Network analysts must be able to see and test the packet data when a malicious network usage is identified and produce actual session by recreating the original data between users as part of forensic analysis. So we need an efficient forensic system to perform this kind of content analysis. The proposed novel technique can be used for content level observation of HTTPS protocol and regenerate original malicious HTTPS session between users for network forensic investigations.*

***Keywords:****Network Forensics, Packet Reordering, HTTPS Traffic Analysis, TLS Parser, Pcap File.*

## I. INTRODUCTION

Network forensics is the study of analyzing network activity in order to discover the source of security policy violations or information assurance breaches [3]. Analysis of the individual traffic flows and their content is essential to a complete understanding of network usage. Many tools let you view traffic in real time, but real-time monitoring at any level requires significant human and hardware resources, and doesn't scale to networks larger than a single workgroup. It is generally more practical to archive all traffic and analyze subsets as necessary. The data stored in the networks can give us a lot of information about the interests, patterns of behavior and even whereabouts of the attacker .This process is known as reconstructive traffic analysis, or network forensics [3]. In practice, it is often limited to data collection and packet level inspection; however, a network forensics analysis tool (NFAT) can provide a richer view of the data collected, allowing you to inspect the traffic from further up the protocol stack. Capturing network activity for forensic analysis is simple in theory, but relatively trivial in practice. Not all the information captured or recorded will be useful for analysis. Identifying key features that reveal information deemed worthy for further intelligent analysis is a problem of great interest to the researchers in the field. Tools like Wire Shark, Packetizer will give enough information for the forensic investigator. But the amount of time that he needs to spend

On these tools is very high for processing large volume of data. So it is often impractical to perform a complete forensic analysis on the entire data due to time constraints and limited human resources. However, good analysis method, visual interfaces and visualizations can vastly improve the time it takes to complete a task, reduce errors, increase concentration and allow a better knowledge of the data. Now day's majority of the network traffic in the internet started using encrypted channels like SSL for communication. Majority of the tools which are performing the analysis of this traffic is doing that in real time, which requires significant human and hardware resources [6] .So a tool is required for performing this analysis in offline mode. A network that has been prepared for forensic analysis is easy to monitor, to trace its security vulnerabilities and configuration problems. In this proposed work, a tool for network forensic investigation of HTTPS protocol is developed which has been successful in recreating the original HTTPS network sessions for tracing any malicious activities. It also allows the best possible analysis of security violations. Most importantly, analyzing a complete record of your network traffic with the appropriate reconstructive tools provides context for other breach-related events. Rest of the paper is organized as follows; section 2 deals with related work, section 3 describes HTTPS traffic analysis process, Section 4 gives detailed working of HTTPS Analyzer, Section 5 provides performance evaluation of proposed tool with existing packet analysis tools, Section 6 gives conclusion followed by references.

## II. RELATED WORK

### 2.1 Network Forensic Analysis

Network Forensic Analysis Tools permits administrators and investigators to monitor networks, gather all information about anomalous traffic, assist in network crime investigation and help in generating a suitable incident response. Forensic tools also provide support in analyzing the inside illegal network event and misuse of resources, predict network pattern in near future, executes risk assessment processes, judging the network

performance, and thus help in protecting the intellectual propriety. An ideal network forensic investigation tools is expected to trace out what exactly happened during a recently past network sessions [5]. This may include replaying of network session, identifying different protocols, sessionizing and auditing the network, locating source of any illegal network activities and monitoring network characteristics etc. Network forensics is being researched for a decade but it still seems to be very young science and many issues like IP spoofing, Mac spoofing etc, are not very clear, ambiguous and found to be still an open problem.

**2.2Existing Network Forensic Analysis Environments**

There exist many network forensic investigation tools which are suitable to investigate different type of protocol and networks. These tools may contain important modules like network monitoring, packet capturing, packet sessionizing, packet reordering, packet reconstruction specific to internet protocols, traffic auditing and evidence manipulation etc. An extensive study on various approaches is made about the development of network forensic investigation tools and corresponding packet reordering methods which are given at the end of this section. Most of these methods are found to be shareware with limited functions, and uses comparatively complex methods for reordering the packets and managing the retransmitted or duplicate packets.

Vicka Corey and Charles Peterman (2004) initially proposed a network forensic analysis tool which addresses how to integrate network security tools and forensic tools to work towards generating valuable evidences in an efficient manner. They presented some security, legal issues and how to manage with forensic tools. They make use of output from security tools, scanners and intrusion detection system as input to forensic section for analysis. They also presented how to sessionize the network capture and parsing of protocols in an efficient manner [3].

Jacob Pennock (2005) proposed an approach for developing remote forensic system to monitor and perform forensic investigation remotely in the network. This proposal mainly concentrates on how data can be collected and preserved for further forensic analysis remotely. This method also addresses execution of forensic analysis in three modes like on demand, periodically and automatic in response to an incident.

Endicott-Popovsky (2007) presents a conceptual framework for embedding digital forensics [11] in the enterprise, where policies, procedures, practices, mechanisms, and awareness training are driven by a system goal expressed as' preserving the ability to prosecute malicious cyber intrusion successfully, while reducing current effort expended on digital forensic investigations that have become too resource intensive. This results in a forensically ready network or network forensic readiness.

J. Scott Haugdahl (2007) proposed an approach towards network forensics which provides detailed case study in identifying many faces of forensics, different possible areas of an organization or network from which probable evidences can be found easily. The proposal includes the psychological, technical, organizational, and contextual factors in developing the forensic tools.

Carol Taylor and Deborah A Fricke (2007) proposed an approach towards digital forensics specification based on forensic policy definition. Their methodology borrows from computer security policy specification, which has accumulated a significant body of research. They first define the process of specifying forensics properties through a forensics policy and then present an example application of the process. This approach lends itself to formal policy specification and verification, which would allow for more clarity and less ambiguity in the specification process.

## III. HTTPS TRAFFIC ANALYSIS PROCESS

HTTP traffic analysis is increasingly hampered by the ubiquitous use of encrypted channels by legitimate and illegitimate network traffic. Both types of traffic are frequently tunneled over application-layer encryption mechanisms, generally using the ubiquitous TLS (SSL) protocol. While it is highly desirable to secure http traffic in this way, it is not an unmitigated benefit as this generally implies that monitoring will be considerably less effective unless special precaution is taken. As a result the traditional network forensics tools are largely limited to recording external characteristics (source and origin addresses and ports, time and traffic patterns), but with little insight into content and purpose of the traffic [8].This section of the tool is dealing with the analysis of the HTTPS traffic in the LAN and to obtain the information from the encrypted traffic that is passing through the network. This section contains two activities for performing this analysis. One is a HTTPS certificate handler (MITM) and the other is HTTPS analyzer. The design of the section is shown in the figure (1).

**3.1 System Design for HTTPS Analysis**

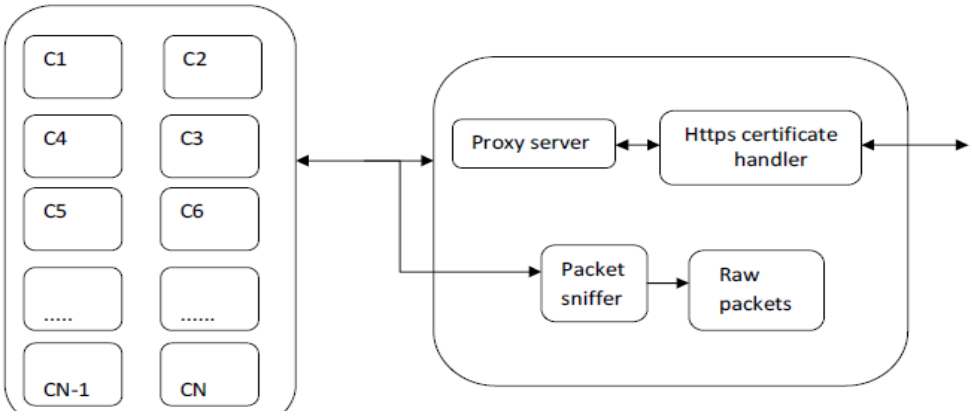


*Fig 1. System Design for HTTPS analysis*

**3.2 Working of SSL**

SSL is an Enterprise Standard adopted by millions of websites to safeguard their online transaction. It ensures the privacy and integrity of transmitted data during the transaction process. Each web server requires one SSL certificate to protect its safety of linkage. SSLscheme uses public and private keys to create an encrypted channel and can be setup at the time of the transaction. Figure (2) shows corresponding SSL Handshake process.

**3.3 SSL Handshake Process**

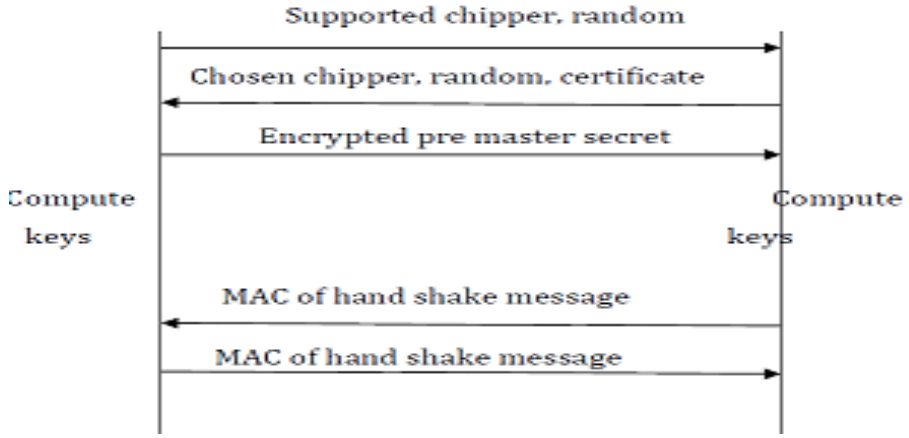


*Fig 2. SSL Initial handshake process*

The connection establishment is as follows

1. The client sends the server a list of the algorithms its willing to support, along with a random number used as the input to the key generation process
2. The server chooses a chipper out of the list and sends it back along with a certificate containing server's public key.

The certificate also provides servers identity for authentication purpose and the server supplies a random number which
is used as part of the key generation process
3The clients verify the server's certificate and extract the server's public key. The client then generates a random secret
string called pre master secret and encrypts it using the server's public key. It sends the encrypted per master secret
to the server

4 The client and server independently compute the encryption and Mac keys from the pre master secret and the client and server random values

5. The client send a mac of all the hand shake message to the server

6. The server sends a mac of all hand shake messages to the client
The main goals of these steps are to agree on a set of algorithm and to establish a set of cryptographic keys. Step 1 and 2 accomplishes the first goal. Second goal is accomplished by step 2 and 3.in step two the server provide the client with its certificate, which allows the client to transmit a secret to the server. After step 3 the client and the server both will share the premaster secret. Step 3 is the key step in the whole handshake. All the data that is going to be protected depends on the security of the pre master secret. The proposed tool willtry to exploit this section so as to decrypt the traffic and perform the web forensic analysis

**3.4 Certificate Handler**

This section is used for handling the SSL certificate and thereby providing facility for HTTPS analyzer to analyze the encrypted SSL traffic. From the above steps of SSL hand shake, it is identified that the MAC key is used for the encryption of the SSL channel. Fromthe step four, it is observed that the Mac keys are calculated using the pre master secret, servers and client's random number from which two random numbers are directly derived. But there master secret in encrypted using the server's public key. So in order to get the premaster secret it needs to have server's private key [1]. In order to decrypt the encrypted premaster secret key value, the methods exploiting the step three of the hand shaking. Here, tool uses a man in the middle type of attack as shown in figure (3) in which the *certificate handler* will be acting MITM attacker. Certificate handler acts as a proxy to the targeted PC/suspect. All traffic from the targeted PCor suspect will be redirected to the certificate handler. Therefore, it can collect the genuine certificate from SSL Server if the targeted PC access to the SSL Server. At the meantime, the certificate handler returns with its own generated certificate, i.e. the certificate handler will intercept the hand shaking process and it will establish two separate SSL connections

- One from the client to the certificate agent
- Another from the certificate agent to the server

In the first connection the certificate agent will use its own fake SSL certificate and the second connection will be using the original certificate from the server

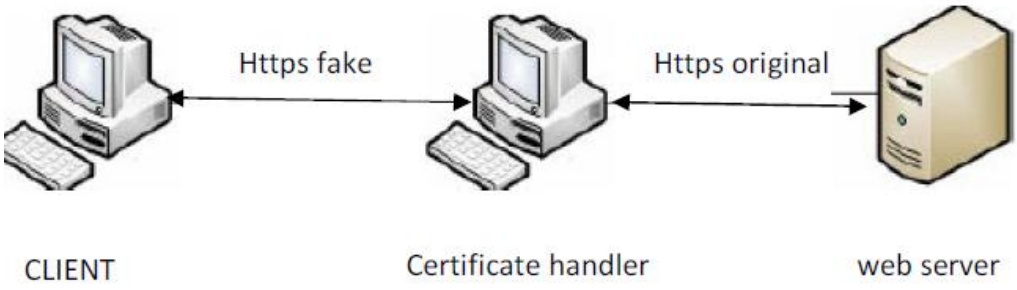


*Fig 3. MITM attack using certificate handler*

The certificate handler will have a root certificate (CA certificate) that is made as trusted in the target host system. For each site the client trying to access through the SSL connection the certificate handler will dynamically create a self-signed certificate with the certificate handlers CA extensions using OpenSSL library using the same private key. As the tool have already made the CA certificate as trusted in the target host the browser will not give any exception. So the user will not get any https error message. The steps for making the root certificate as trusted in the target host are given in the next section.3.5 Adding the Root Certificate to the Trust Store in the Target HostSetup a Local Certificate Management Console:

- Start, Run: mmc
- From the Microsoft Management Console:
- File, Add/Remove Snap-in…
- Click Add…
- Select Certificates and click Add
- Make sure My User Account is selected and click Finish
- Again, make sure Certificates is selected and click Add
- This time select Computer Account and click Next
- Make sure Local Computer is selected and click Finish
- Click Close

- Click OK

Now we have a Management Console that looks like as figure (4).

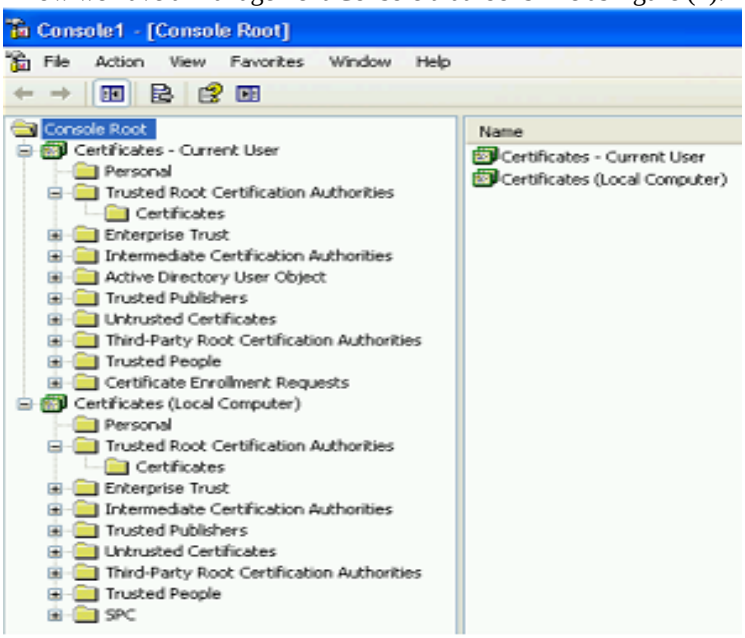


*Fig 4. Certificate management console*

You can now save this:

- File, Save
- Enter a name: Local Certificate Management
- Click Save
- Copy the root certificate (CA certificate)to the target system
- Open the Local Certificate Management Console that we setup earlier.
- Open Certificates – Current User, Trusted Root Certification Authorities, Certificates
- Right click in the Certificate Area on the right and choose All Tasks, Import…
- Browse to the root certificate file
- For Certificate Store, make sure place all certificates in the following store is selected
- with Trusted Root Certification Authorities
- Click Yes to the security warning
- We can now see this certificate in our Personal Trusted Root Certification Authority
- Store.

Now this certificate is added to the trust store in the target computer. So it will not give anyhttps error while the certificate handler is performing the MITM attack.

## IV. WORKING OF HTTPS ANALYZER

This is the main section of HTTPS analysis process. HTTPS analyzer works on the raw packets that are collected from the proxy system and perform https analysis on those packets. The main goals of this section are to calculate the session key used for the traffic encryption, decrypt the encrypted HTTPS traffic and produce the forensic result of the traffic. The session keys are calculated on the basis of the initial handshake messages that are used for establishing the SSL session between the client and the server [1]. In order to calculate the session key, section needs to have the private key of the certificate used for the SSL connection .So only the legitimate users who are having access to private key can perform this task. The following flow diagram will give an idea about the steps followed by the HTTPS analyzer for performing the HTTPS forensic analysis.

### 4.1. Work Flow Diagram for Https Analyzer

#### 4.1.1 Pcap File

This file consists of all the Ethernet packets that we have captured from the Ethernet card. This can be done by using the capture section in our software or using some other software's like Wire shark that support Pcap format. The analysis process is performed on this file. This can be done either using our software in the system in which the analysis is to be performed or by capturing the packets using some other software and

export that file to a system that contains our software. If the tool needs to get all the packets that are passing through the Ethernet card then the tool must perform the capture in promiscuous mode.

**4.1.2. Filter HTTPS Packets**

This is done on the basis of the port that is set on the proxy for the HTTPS traffic. The tool will filter those TCP packets that having either the source port or the destination port as the HTTPS port

**4.1.3. Separation of each Section**

In order to perform the analysis process, the tool separates each connection that is created for the http traffic. This can be done using the combination of the source port and the destination port

**4.1.4 Reordering the packets**

Due to the multipath routing or parallelism at the routers, the packets that we have collected in the Pcap file may be out of order. So in order to perform the analysis, this section reorders those packets. The algorithm is given below briefly as Work flow diagram of HTTPS analyzer. It consists of two phases. First four steps constitutes first phase and second phase starts at fifth step onwards.

**4.1.5 Work Flow diagram of HTTPS Analyzer**

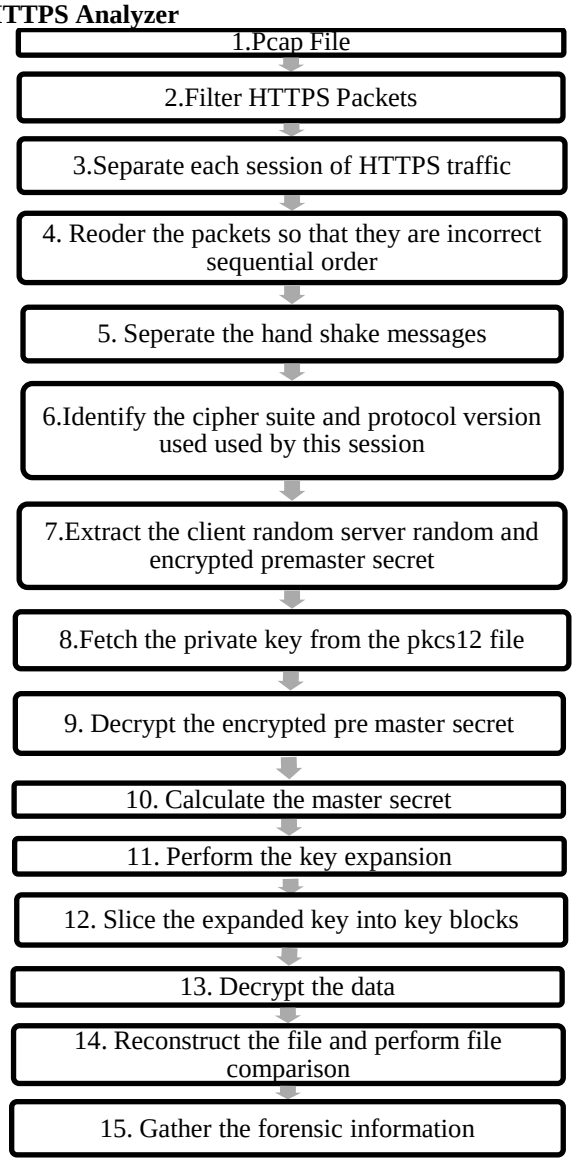


*Fig.5 Flow diagram of HTTPS Analyzer*

**4.1.6 Separating the hand shake messages and extracting required fields**

This section needs to collect the handshake message from client to server and from server to client. This is done using a TLS parser. The design of TLS parser will be explained later. Using this TLS parser, the tool extracts the protocol version, cipher suite, client random, server random and the encrypted premaster secret. These five fields are required for the key generation process and the decryption of the traffic [4]. In order to extract these fields, the tool needs to parse three types of handshake message Client hello, Server Hello, Client key Exchange .Details of these messages are given below

**4.1.7 Client Hello Message**

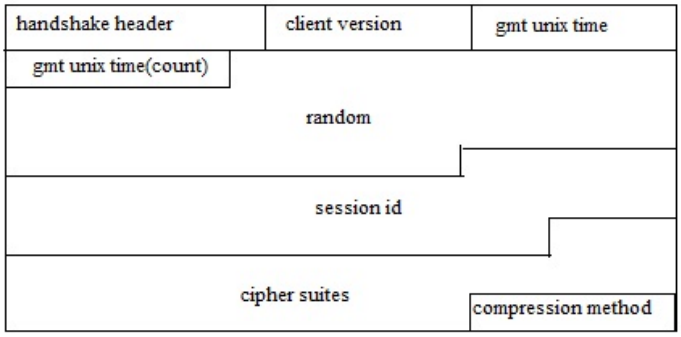


*Fig 6. Format of Client Hello Message*

The client hello message as shown in figure (6) is sent when a client connects to a server to initiate the handshake. After the common handshake header is the client version, which specifies the highest version of the protocol that the client supports. SSL and TLS Implementations need to support all previous versions of the protocol as well. Then comes a random number which consists of the current GMT time-stamp and 28 bytes generated by a cryptographically secure pseudo number generator. After the random number comes the session id. If supplied, this can be used to perform an abbreviated handshake sequence by reusing key material that was previously negotiated between the client and the server. Next comes the list of cipher suites that the client is prepared to use. Cipher suites define the encryption and hashing functions that will be used by the connection once the handshake concludes as well as the key exchange method used during the handshake. They are represented by two 8bit numbers which is documented in the SSL or TLS specification or by some additional specification. For example TLS RSA WITH RC4 128 SHA is a cipher suite that uses RSA for key exchange, RC4 as the bulk encryption algorithm, SHA1 as the hashing function for the MAC. From this client hello, the tool extracts the client random number.

**4.1.8 Server Hello Message**

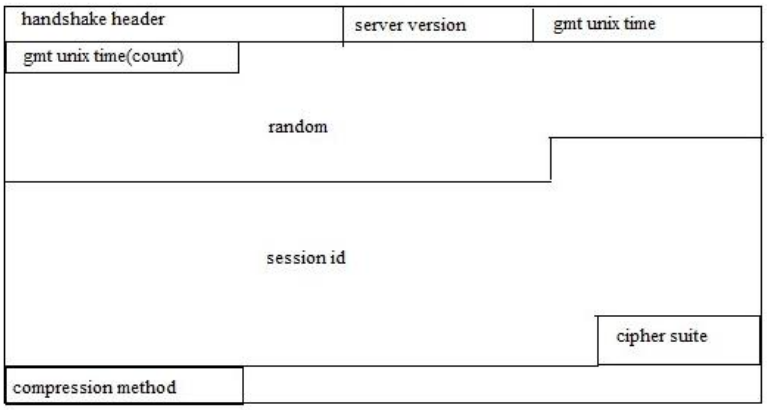


*Fig 7. Format of Server Hello Message*

The server hello shown in figure (7) is sent by the server in response to a client hello. The server version is the client version if it is supported by the server, else the highest version supported by the server. Then there is a random number consisting of the GMT time-stamp and 28 bytes generated by a cryptographic random number generator. Next is the optional session ID. If this is the same as the session ID sent by the client in the client hello then an abbreviated handshake will be performed using key material cached by both the client and the server. If empty, it indicates that the server is not willing to perform an abbreviated key handshake. If the session ID not empty then tool cannot decrypt the traffic because the traffic is using the cached key material. Finally the message includes the cipher suite and compression method selected by the server from the lists provided in the client hello. From this message, tool extracts the protocol version, cipher suite and the server random number.

**4.1.9 Client key exchange**

Th3 client key exchange message format is shown in figure (8) is sent immediately after the client certificate is sent. It has two variants, one for when RSA is used as the key-exchange algorithm, and one for when Diffie-Hellman is used. Here the tool uses RSA. When RSA is used, the client key exchange message consists of the header handshake followed by PKCS #12 encoded pre-master secret that is encrypted using the server's key, as sent in the server certificate message. The handshake header is not encrypted. The pre-master secret consists of the maximum version supported by the client followed by 48 random bytes generated by a cryptographic random number generator. Here the tool extracts encrypted premaster secret.

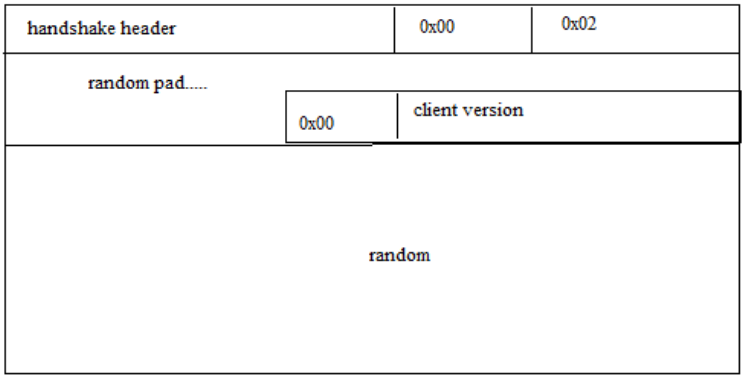


*Fig8 Client key exchange packet format*

**4.1.10. TLS Parser Overview**

A TLS parser is developed for separating the fields in the HTTPS connection so that the tool can retrieve the basic information about the connection and the data required for the decryption on the HTTPS traffic. The basic functionality of the TLS parser is to retrieve features from the data like server random number, client random number, cipher suit, protocol version, encrypted application data, encrypted premaster secret [4]. The TLS parser will first convert the data in to corresponding hex value .This hex value is parsed in order to get the details. The structure of the parser is shown in the figure (9) below.

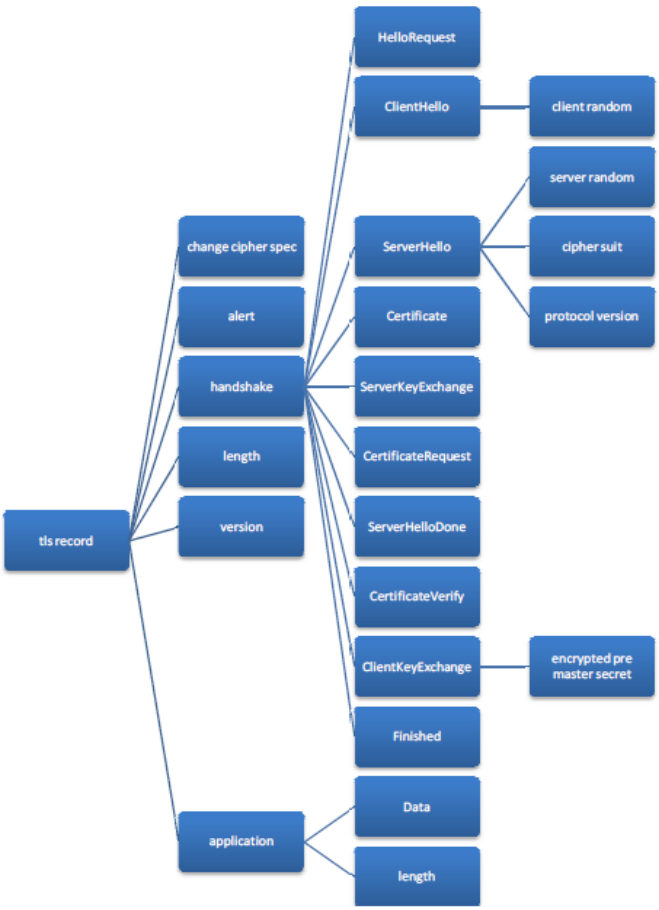


*Fig 9. TLS Parser Design*

**4.1.11 Decryption of the premaster secret**

As already specified in the third step of the SSL handshake the pre master secret is encrypted using the server's public key. As the key with us in the pkcs12 file, the tool fetches the private key from the file. At this point, tool will identify the algorithm used for encryption of the premaster secret from the cipher suite and perform the decryption operation of the encrypted premaster secret using the specified algorithm and obtain the original premaster secret.

**4.1.12 Key Material Generation**

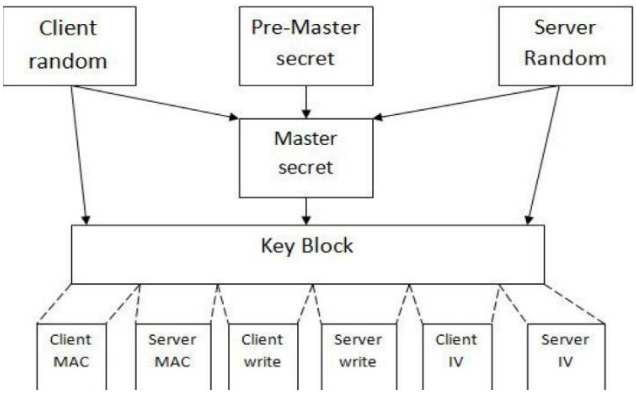


*Fig10. Key Derivation*

Key material generation as shown in figure (10) is used to generate the secret material that will be used as keys and input vectors to encrypt and verify records. The inputs to key generation are the client and server random, and the pre-master secret, sent in the client and server hello, and client key exchange messages respectively. The first step in the key material generation is to transform the pre master secret in to the master secret. This is done by applying the pseudo random function PRF to the pre master secret, client random and server random: Master_Secret=PRF (pre_master_secret, "master secret" ,client_random + server_ random)[0..47]. Now the tool uses the PRF with the master secret to obtain the key block. Key_block=PRF (Master_Secret, "key expansion", server_ random +client_random). The key block is as many bytes long as is needed and is divided up into MAC secrets, symmetric encryption keys, and input vectors for blocking used with symmetric encryption [11]. These are, in order, the client write MAC secret, server write MAC secret, client write key, server write key, client write IV and server write IV. Unneeded values are omitted.

**4.1.13 Pseudo Random Function (PRF)**

Pseudo Random Function (PRF) is at the heart of key material generation. It takes three arguments, a secret a fixed ASCII string and a seed as in PRF (secret, label, seed). First the secret part is split in to two halves S1 and S2.Each half is used as a secret component in an expansion function P_Hash which is based on HMAC. P_hash generates an arbitrary length byte string by the process shown below.

P_hash(secret,seed)=HMAC_hash(secret,A(1),A(0))+
HMAC_hash(secret,A(2),A(0))+
HMAC_hash(secret,A(3),A(0))+….
A(): A(0)= seed
A(i)=HMAC_hash(secret,A(i-1))
HMAC_hash means HAMC using a given hash algorithm,e.g..,
HMAC_MD5 means HMAC using MD5. This is run as many times as desired. So to produce 48 bytes of output, it would be run 3 times for MD5, and for SHA-1 it would also be run 3 times and the last 12 bytes of output would be discarded. Now PRF is constructed by XORing a P_MD5 and a P_SHA-1:
PRF (secret, label, seed) =P_MD5 (S1, label + seed)⊕ P_SHA-1(S2, label + seed); [11].
14) SSL v3 key derivation
SSL v3 Key derivation is similar to TLS key derivation, except that the PRF is replaced with a series of expansion functions based on a combination of MD5 and SHA-1[11]. The use of constants "A", "BB", etc. ensures that the output of each digest is different, even though the secret data is the same. The process is shown below Master Secret = MD5 (pre master secret + SHA1 ("A"+ pre master secret +client random +server random)) + MD5 (pre master secret + SHA1 ("BB"+ pre master secret +client random +server random)) + MD5 (pre master secret + SHA1 ("CCC"+ pre master secret +client random +server random)).
Key_Block= MD5 (Master Secret + SHA1 ("A"+ Master Secret + server random +client random)) + MD5 (Master Secret + SHA1 ("BB"+ Master Secret + server random +client random)) + MD5 (Master Secret + SHA1 ("CCC"+ Master Secret + server random +client random)) +….

**4.1.14Decryption of the application data**

Final step of the HTTPS analysis is the decryption of the application data using the key blocks that is being generated in the previous step (Key material derivation) [11]. At first tool separates the traffic in to two parts one coming from the Server (certificate handler) and the other goingto the Server (certificate handler).Now these two section are send to the TLS parser separately. The TLS parser will parse the application data messages and fetches the data part from it and write it to a file. Now we will be having two files one containing encrypted data from server to client and the other containing encrypted data from client to server. Now this section performs the decryption on these two data files with the appropriate keys. If the file contains the data from server to client then, the tool uses the server write key and the server IV for decryption otherwise we will use client write key and the client IV for decryption purpose [7].

4.1.15 Reconstruction of data and Comparing the File

In this step, the tool is reconstructing the data by separating the header and data field form the decrypted data that have been passed through the network and identify the file type of that data and save them in the specified folder in the hard disk. Now this reconstructed file is compare with the given file. If a match is found then it will be reported in the forensic details section.

**4.4.16 Gathering and displaying the forensic details**

Once the original session is reconstructed, the tool will trace the forensic details of the HTTPS data that has been reconstructed in the packet and display those details in the GUI so that the user can understand what has happened to that system during a particular network session. Forensic details are displayed in four separate windows for the easiness of analysis. One window contains the header details which will give the idea about the HTTP headers that has been transferred in those connections. The second window will give the details of each packet with its time of arrival or departure source and destination IP, Mac ID, and ports. The third window will display the hex code of the data that has been reconstructed. The hex code represents the pattern of the data which can be used by other pattern matching software's for further analysis purpose. The fourth window gives an over view of the total no of files created its location its file path file type and length. The fifth window will display the in-depth forensic details of each file. It contains the file name, file type, the length of the file, the time at which the file is downloaded or uploaded, the source mac address, source IP, source port, the destination mac address destination IP, destination port etc. Thus the aim of proposed work is to identify the system and then to handover the case to police with exact malicious network stream reconstructed (proof) for further investigations.

## V. PERFORMANCE EVALUATION OF PROPOSED TOOL WITH EXISTING PACKET ANALYSIS TOOLS

A brief comparison on the performance of packet stream reassembly of proposed tool with other freely available packet analysis tools is done in this section. The proposed tool is named as Network Forensic Analysis Tool Kit (NFATK). The other tools which are considered are not purely used for forensic analysis, but used for packet analysis or session analysis. Some tools give session reconstruction facility. Figure (11) provides time taken to reconstruct HTTPS streams with average number of packets in seconds and proves that the proposed method is much efficient in regenerating corresponding HTTPS stream.

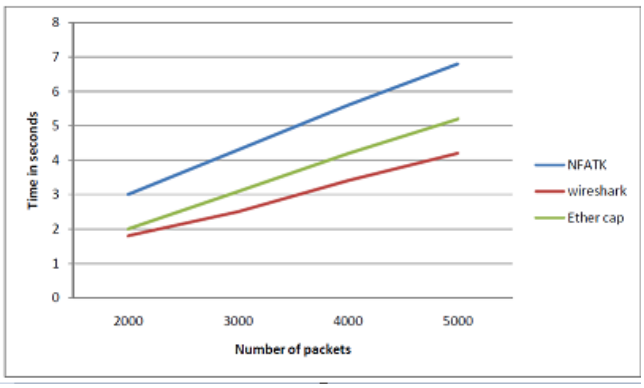


*Fig11. Time Analysis for HTTPS Reconstruction.*

## VI. CONCLUSIONS

The proposed methodology is a strategic approach for the session recreation of HTTPS analysis as part of network forensic process and further related investigation. Thus this forensic analysis would recreate all transactions through HTTPS protocol to check whether it is malicious or not. This tool can be effectively used by network administrators, forensic investigators to identify any kind of network traffic breaches or sessions based on above mentioned protocol and to perform internet or network traffic content analysis.

## REFERENCES

[1] Alberto Ornaghi, Marco Valleri (2003). "Man in the middle attack demos." black hat conference USA.
[2] E. Anderson, M. Arlitt (2006). "Full Packet Capture and Offline Analysis on 1 and 10 Gb/s Networks" HP Laboratories Palo Alto HPL-2006-156
[3] Vicka Corey, Charles Peterman (2002). "Network Forensics Analysis" IEEE INTERNET COMPUTING
[4] Eric Rescorla (2006). "SSL and TLS Design and Building Secure Systems." Addison Wesley.
[5] Ka-CheongLeung, Li, V.O.K.Daiqin (2007). "An Overview of Packet Reordering in Transmission Control Protocol (TCP): Problems, Solutions and Challenges." IEEE Transactions On Parallel And Distributed Systems.
[6] Kefei Cheng, MengGao, RuijieGuo (2010). "Analysis and Research on HTTPS Hijacking Attacks." Second International Conference on Networks Security, Wireless Communications and Trusted Computing.
[7] Moxie Marlinspike (2009). "New Tricks For Defeating SSL In Practice." Black hat conference DC
[8] Natarajan Meghanathan, Sumanth Reddy Allam (2009). "Tools And Techniques For Network Forensics." International Journal of Network Security & Its Applications (IJNSA), Vol .1
[9] Seung-hoon Kang, Juho Kim (2008). "Network Forensic Analysis Using Visualization Effect." IEEE International Conference on Convergence and Hybrid Information Technology
[10] Barbara E. Endicott-Popovsky ,Deborah A. Frincke (2007) "Embedding Forensic Capabilities into Networks: Addressing Inefficiencies in Digital Forensics Investigations." IEEE Workshop on Information Assurance United States Military Academy, West Point, NY
[11] ThawatchaiChomsiri (2007). **"**HTTPS Hacking Protection." IEEE Advanced Information Networking and Applications Workshops (AINAW)