Proceedings of the 2023 International Conference on Software Engineering and Machine Learning
DOI: 10.54254/2755-2721/8/20230149

A survey on packet capture: Tool introduction and security vulnerability description

Yuxuan Wu[1]**, Tuosheng Jiao**[2,3]

[1]School of Materials and Metallurgy & Computer Sciences, Wuhan University of Science and Technology, 430065, China, wuyuxuan8398@163.com
[2]Manning College of Information & Computer Sciences, University of Massachusetts Amherst, 01003, US, tuoshengjiao@gmail.com
All the authors contributed equally to this work and should be considered as co-first authors.

[3]tuoshengjiao@gmail.com

Abstract: This work involves discovering network security vulnerabilities based on existing packet capture software. By learning the operation principle of packet capture software, we discuss the hidden network security problem among them. There are two basic packet capture software mentioned —— Wireshark and Sniffer. Analyse the SSL (Secure Socket Layer) protocol and HTTPS (Hypertext Transfer Protocol Secure) protocol to observe where causes a vulnerability in those protocols that allows the thief to obtain information through packet capture software. Finding an attack mode of the thief, we further study and discuss how to reduce the risk of information leakage caused by vulnerability existing. We hope the perspective proposed is not only to avoid workstation chaos caused by packet capture software as far as possible, but also to further encrypt the plaintext data to make it more difficult for packet capture tools to parse the packet.

Keywords: network security vulnerability, SSL, HTTPS, packet capture, wireshark, sniffer.

1. Introduction

Packet capture is the operation of intercepting, resending, editing, and storing data packets sent and received by network transmission. Modern packet capture technology is often used for data interception and network security detection. Data is transmitted over the network in small units called frames. The frame is made up of several parts, different parts with different functions. Frames are formed using a specific software called network drivers and then sent to the network cable via a network card. They travel through network cables to the target machine. The opposite process is executed on one end of the target machine. In general, all devices on the network can "listen" to traffic passing through, but they do not respond to packets that do not belong to them. For example, workstation A does not capture data that belongs to Workstation B, but simply ignores it. If the network port of the workstation is in mixed mode, all packets and frames on the network can be captured.

1.1. Tool
With packet-capturing technology, packet-capturing software is emerging. There are many popular packet capture tools like Wireshark and Sniffer. Although they have different functions, their basic principle is the same. Our computer transmits data through the network by uploading and downloading some data packets from the network. Usually, the software that sends or receives these data packets will process them. Ordinary users do not care, and these data packets will not always be saved on the user's computer. Packet-capturing tools can help us keep these data packets. If these data packets are transmitted in clear text, or we can know their encryption methods, then we can analyse the contents of these data packets and their purposes [1].

1.1.1. Wireshark. Wireshark (formerly known as Ethereal) is one of the most widely used network packet analysis software with potent functions. It can capture and display various types of data packets in real-time, filter data packets in various ways, and perform various statistical analysis [2].

1.1.2. Sniffer. Sniffer, as a sniffer for listening and analysing the network. It can be used to monitor the status of the network and the information transmitted on the network. Hackers often use sniffer technology to intercept users' passwords. But if Sniffer technology is applied to network fault diagnosis, its powerful network analysis ability will become a sharp tool for network maintenance personnel to quickly locate network fault points [3]. The Sniffer program is a tool that takes advantage of Ethernet features and sets net cards (generally Ethernet cards) to promiscuous mode. Once the network card is set to this mode, it can receive every packet transmitted over the network. Normally, the network adapter receives only the packets associated with its address, that is, the packets sent to the local computer. To enable the Sniffer to receive and process information in this manner, the system must support BPF, and SOCKET-PACKET in Linux. In general, however, network hardware and TCP/IP stacks do not support receiving or sending packets not associated with the local computer. Therefore, to bypass the standard TCP/IP stack, the network card must be set to the mixed mode we discussed. To activate this method, the kernel must support the pseudo-device Bpfilter; root permission is required to run the program. Therefore, the sniffer needs to be installed as the user root. If you only enter the system as a local user, sniffing the root password is impossible, so Sniffer cannot be run [4].

1.2. Security vulnerability
Hundreds of security problems are waiting to be improved in HTTPS and SSL protocols. Attackers can use middleman attacks to achieve more easy attacks. For example, when the attacker first attacks the middleman, he modifies the HTTPS link through the HTTPS vulnerability to redirect the user to the malicious HTTPS website. After the middleman attack succeeds, the attacker can launch two dangerous attacks: (1) Use a cookie tampering attack. This attack is when the attacker uses the browser to constantly change the cookie during the user's session and repeatedly marks the same cookie as valid. If the attacker can hijack the cookie from the website in advance and implant it into the user’s browser, when the user's authentication information reaches the HTTPS site, the attacker can obtain the user's credentials and log in as the user; (2) Redirection attack. The attacker can inject a JavaScript script into the URL and modify the new tab to direct the attacker to a malicious login page. (3) key exchange algorithm spoofing.

2. HTTPS protocols

2.1. Symmetric and asymmetric encryption
First, a simple concept is that HTTPS is an upgraded version of HTTP. When referring to HTTPS, one must know two encryption concepts.

The first encryption concept is the concept of symmetric encryption. Simply put, symmetric encryption means that the client and the server use the same key, and both the client and the server can use this one key for encryption and decryption simultaneously. The advantage of this encryption method is that the encryption is very efficient. However, since the key is placed on the client's side, there is a high theft risk. Once the key is stolen, it will result in all the information encrypted by the key being stolen or even modified.

Another encryption method is asymmetric encryption. Asymmetric encryption, as the name implies, means that both sides are different. In this case, it is divided into two keys. One is the public key, and the other is the private key. The public key is placed on the client, and the private key is placed on the server. The public and private keys are in one-to-one correspondence [5]. For example, Bob makes a lock that has only one key in the world and belongs to Alice only. Only Alice can unlock Bob's lock in the world. (Excluding other methods such as brute force unlocking). Thus, the content encrypted by the public key can only be decrypted by the corresponding private key. At the same time, the content encrypted by the private key can only be solved by the corresponding public key. The advantage of this encryption method is high security, but the disadvantage is also apparent: the efficiency of encryption and decryption is too low.

2.2. HTTP protocols

As mentioned earlier, HTTPS is an upgraded version of HTTP, and the next is the basic concept of HTTP. For the current technology, HTTP is no longer a secure way to transmit information. As shown in Figure 1: In HTTP, all data will be transmitted in plaintext [6]. Thus, once an eavesdropper accesses the message, they will directly know what is being sent. For example, if Bob writes a love letter to Alice, and Bob writes all the information directly in the letter, then maybe the letter will be intercepted by Justin, and Justin will know some secrets about Bob and Alice. However, at this point, what Bob wrote in the letter was encrypted content. When Justin intercepts the letter, he still cannot read the contents of the letter, which significantly increases the security of the message delivery. This method of encrypting messages is also known as HTTPS.

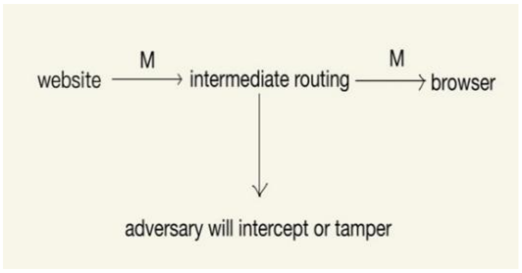

Figure 1. HTTP protocol transmission diagram.

2.3. HTTPS protocols

Because of the large amount of data that needs to be exchanged in the daily exchange of information and to keep the information secure. Hence, HTTPS is formed by two methods working together: symmetric encryption and asymmetric encryption. The browser and the website need to develop a key first, and then they exchange information through the key using the symmetric encryption method. Since the browser and the website have not been exchanged beforehand, it is difficult to exchange keys securely the first time [7]. In this case, HTTPS uses asymmetric encryption. First, the browser generates a key A, then encrypts it with the public key PK (how to get the public key PK will be described later to create the cipher text C (C=A+PK). When the website has C, it can decrypt it with the private key prepared beforehand to get the value of A. This process is also known as simple asymmetric encryption. This process is also known as simple asymmetric encryption. HTTPS will use this encryption method for the first communication [8].

2.4. Digital signature

In the first transmission process, how the browser gets the public key PK of the website is also a critical point. Before solving this problem, a new digital signature concept needs to be introduced.

Because during the transmission of information, the enemy can easily steal or modify the data. Therefore, when we receive a message, it is essential to determine whether it has been tampered with. In real life, people would look at the handwriting or directly at the signature on the back, or more intuitively, to see signs of alteration. But in the world of the Internet, it is difficult to achieve this. Therefore, there is a digital signature, meaning the original encrypted data is encrypted again. When the recipient receives this message, he will use the digital signature algorithm to decrypt it together with the original text. Only when both messages have not been tampered with can we receive a message that has not been tampered with? If the letter has been tampered with, the recipient will find it impossible to decrypt the news [9].

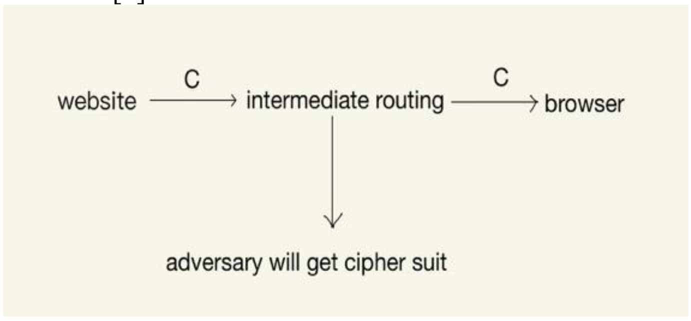

Figure 2. HTTPS protocol transmission diagram.

2.5. CA Institution certificate

Here, a new concept is introduced: the CA authority, which issues digital certificates to each website. When the browser receives a PK with a CA certificate, it gets an accurate and reliable public key PK. The specific process is that first, the website needs to apply to the CA organization and then send its PK to the CA organization, which will encrypt the PK together with the website's domain name, a good duration, and a series of important information to make a certificate, and then send the certificate to the website after encrypting it with the CA organization's private key [10]. The certificate will then be encrypted with the CA organization's private key and sent to the website. When the browser applies for the public key from the website, the website will send the certificate obtained from the CA organization to the browser. If the browser decrypts the certificate with CA's public key, it can get the certificate issued by CA and the website's private key if the decryption is successful. If the decryption fails, the public key may have been tampered with [11].

The first question is how the browser obtained the CA's public key. The second question is whether the CA authority is really safe and reliable. First, since only a few CA organizations exist, the open operating system will have the public keys of the mainstream CA organizations built into the system. When decrypting, you only need to go through all the built-in public keys as long as one succeeds. As mentioned earlier, when CA authorities create certificates, they will add other data, such as website domain names to assist in verification. When the website sends the public key to the browser, the browser will decrypt it because the CA authority will add the domain name and other information to the certificate, so the browser will check whether the received public key and the domain name in the CA authority are consistent, and terminate the request if they are not [11].

3. SSL principle

3.1. Handshake protocol

The following section describes some of the principles CA authorities use in making certificates. Here the SSL protocol is involved. the SSL protocol is located between the application layer and the TCP layer. There are three main purposes: (1) The client confirms that the server is a real and secure server. (2) The server confirms that the client is a real secure client and is not impersonated by a hacker. (3) To establish a data channel between the server and the client. SSL protocol consists of three main protocols. The first is the handshake protocol. The handshake protocol is used when the client and server communicate with each other using SSL. It includes a series of messages between the client and the server. The client and server need to authenticate with each other to communicate properly. There are four main phases in the handshake protocol of the SSL protocol.

The first phase: As shown in Figure 3, the client sends a hello to the server, which includes version, random, session id, cipher suit, and compression. The server confirms the contents of the hello, and if the confirmation is successful, the server enters the second phase [12].

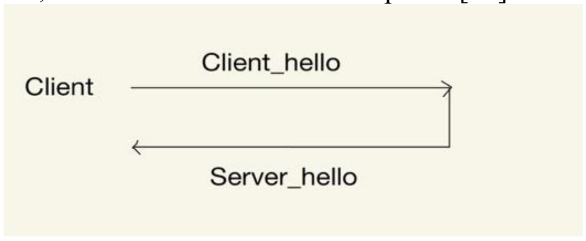

Figure 3. Schematic of the first phase of the handshake protocol in SSL.

The second stage is shown in Figure 4, in the second stage the server sends to the client unilaterally, the server is the sender and the client is the recipient. In the second stage, the server will send four things to the client. (1) Certificate: The certificate carries the server's public key, which lets the client verify that the server is accurate and secure. (2) Perform a key exchange. (3) Certificate request, which requests the client to send its certificate, used by the server to verify the authenticity and security of the client. (4) End: marks the end of the second phase and the beginning of the third phase [12].

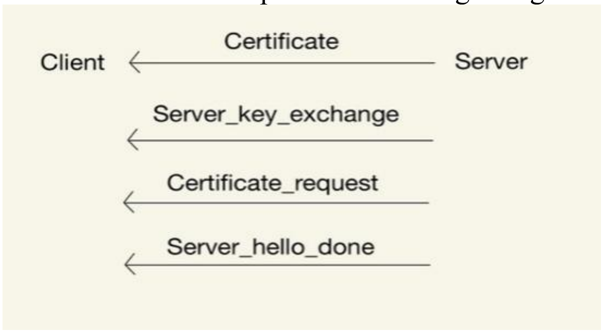

Figure 4. Schematic of the second phase of the handshake protocol in SSL.

The third phase is shown in Figure 5. In the third phase, the client sends information to the server unilaterally. In the third phase, the client sends three messages to the server. (1) Certificate: This certificate is used to verify its identity. (2) Key exchange: The client encrypts its private key with the public key sent from the server and then sends it to the server. (3) Certificate verification: Sign the prepared secret and random number to prove itself [12].

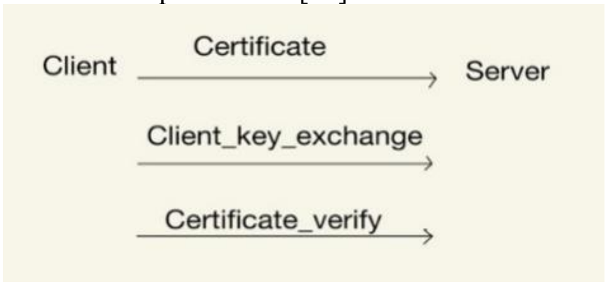

Figure 5. Schematic of the third phase of the handshake protocol in SSL.

The fourth phase is shown in Figure 6, where the client requests to update the cipher suite and subsequently sends a message using the algorithm, the key obtained in the previous phase, expressing the end of key exchange and authentication. After that, the server updates the cipher suite and sends a message expressing the end of the exchange. The above is about the first protocol in SSL protocol [12].

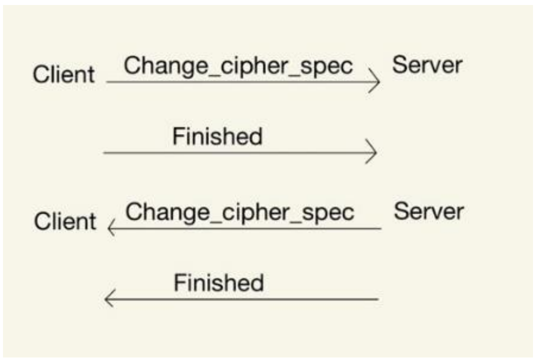

Figure 6. Schematic of the fourth phase of the handshake protocol in SSL.

3.2. Logging protocol
The second protocol is the logging protocol, which has two main functions. (1) The record protocol encrypts and transmits the data using the algorithm previously determined in the handshake protocol. (2) The message authentication code is calculated using the MAC key with points in the handshake protocol [13].

3.3. Alert protocol
The third protocol is the alert protocol. When a client or server finds some vulnerability or error in the communication, they will notify each other. If the vulnerability or error is significant, they will shut down the SSL protocol to ensure the security of the message [13].

4. Vulnerabilities and means of attack
However, such a secure protocol can also have vulnerabilities that can cause information leakage if they are discovered by someone interested. Several common vulnerabilities and attacks are described below.

The first one is based on a man-in-the-middle attack. It is also called cookie tampering attack. Because in daily life, the client and the server are not under the same network, when the client requests the server, it will be forwarded through a nearby gateway, and the client will send the request to the gateway, and then the gateway will send the request to the server. In the process of relaying the gateway, the hacker will impersonate the gateway by using a forged private key and certificate prepared in advance to impersonate the gateway. In daily life, man-in-the-middle attacks are common in public wife and public routing, to avoid man-in-the-middle attacks, people should try to avoid using public networks, and even if they do, they should pay attention to some information alerts [14].

The second type is a communication business flow attack and a redirection attack. Hackers use the length of the ciphertext information to infer the information about the URL through a series of technical means to get what website the client is visiting. Because the length of the ciphertext reveals the length of the plaintext information, the hacker can take advantage of it. At the same time, the SSL protocol only has a padding mechanism for packet cipher algorithms and does not support stream ciphers. However, communication service stream attacks require more prep work and too many prerequisites, it is not very harmful, but it is still a vulnerability that cannot be ignored by the SSL protocol. In daily life, people need to pay more attention to the protection of their information to avoid being exploited by unsuspecting people [14].

The third type is key exchange algorithm spoofing. In the second stage of the handshake protocol mentioned above. The server and the client will perform a key exchange. Various keys such as the RSA algorithm and Diffie-Hellman are used in the exchange process, but the exchange is not in the public signature. Thus, the hacker can use the DH parameter signature to trick the user and use cipher suit. to get all the information before sending the end of the call in the fourth phase. Since the master key of the client is lost, it leads to the fact that all the messages after that will be forged [14].

5. Discussion

This paper explains the HTTPS protocol and SSL handshake protocol in detail. We summarize three types of the current security vulnerabilities after deeply studying HTTPS protocol and SSL protocol; (1) use cookie tampering attack; (2) communication business flow attack and redirection attack; (3) key exchange algorithm spoofing. HTTPS protocol and SSL handshake protocol still have many places to be further improved, for example, how to prevent one person from being used by other hackers when attacking the client. Even if the information leakage caused by the man-in-the-middle attack is not serious, the hacker who attacks through the man-in-the-middle will forge the digital signature certificate or private key to hide himself, which may lead to greater information leakage. We propose that people should pay more attention to their personal information, especially to avoid revealing private information under public networks. Protocols need to further upgrade and reinforce themselves to avoid more and more accurate attacks by hackers. After all, attackers can achieve many easier attacks by using man-in-the-middle attacks. You have to perform man-in-the-middle attacks and are forced to become a very determined attacker. This is a devastating disaster for e-commerce.

Author’s contributions

Yuxuan Wu and Tuosheng Jiao designed research, performed research, analysed data, and wrote the paper. Their contributions are equal.

Ackonwledgments
Yuxuan Wu and Tuosheng Jiao contributed equally to this work and should be considered co-first authors.

References

[1] CHEN Yi-kuang. Analyses on some anti-virus software by Sniffer software[J]. Computer Knowledge and Technology, 2014,10 (25): 5869-5872. DOI: 10.14004/j.cnki.ckt.2014.0248
[2] Zhao Yi. Application Case of Using Wireshark to Realize Packet Analysis [J]. Computer Programming Skills and Maintenance, 2018,0 (5): 106-108. DOI: 10.3969/j.issn.1006-4052.2018.05.040
[3] Li Linfeng. The application of sniffer tool in network management [J]. Henan Science and Technology, 2013,0 (11): 18-18. DOI: 10.3969/j.issn.1003-5168.2013.11.015
[4] Liu Botao, Zhao Gang, Feng Cuili, Tang Le. Sniffer principal analysis and Win cap implementation [J]. Journal of East China Jiaotong University, 2005, (05): 102-105
[5] Li B. An introduction to asymmetric encryption methods and their applications [J]. Information Recorded Materials,2021,22(01): 214-215. DOI: 10.16009/j.cnki.cn13-1295/tq.2021.01.148.
[6] Qiu-Ling Yang. Sniffer-based HTTP analysis [J]. Fujian Computer,2012,28(04):168-169.
[7] Guo Haoran. Analysis of the advantages and disadvantages of HTTPS for website security [J]. Computers and Networks,2017,43(05):50-51.
[8] Sun Yuhang. An introduction to the principle and role of "https"[J]. Computer Products and Circulation,2020(01):132.
[9] Huang Shuo. Application of RSA digital signature algorithm in software encryption[J]. Network Security Technology and Applications, 2018, No.210(06):37+52.
[10] Zhang Taotao. HTTPS security discussion [J]. China Science and Technology Information, 2021, No.645(Z1):73-74.

[11] Xu Yuantao, Zhou Ning. Design and implementation of CA of certificate issuing authority[J]. Computer and Telecommunication,2007(04):47-50.
[12] Tan Yale, Hu Ximing, Ma Miao. Exploration of the working process of SSL protocol[J]. Network Security Technology and Applications, 2017, No.199(07):36-38.
[13] Zhang Baoyu. An analysis of the principle and application of HTTPS protocol [J]. Network Security Technology and Applications,2016(07):36-37+39.
[14] Hu Guohua, Yuan Shujie. Analysis of SSL protocol security flaws [J]. Computer System Applications, 2006(08):94-96.