CYBER ATTACKS EXPLAINED: CRYPTOGRAPHIC ATTACKS

Cryptographic solutions are used to encrypt data transmission over wireless or wired protocols. Unfortunately, these techniques have proved to be vulnerable to attacks and data can be stolen. This article explores the various means to strengthen encryption techniques to protect network infrastructures including methods using FOSS-based solutions.

As we all know, the heart of cryptographic network communication is public-key cryptography (PKI), which is used to encrypt the TCP/IP communication between two network end-points. PKI uses various encryption algorithms to ensure data security. The whole idea behind encryption is to make it a very difficult, time-consuming task to try out all possible keys. For example, if a message is encrypted using an 8-bit key, it means that 256 different combinations of the key need to be tried to decrypt the data. Any computer can perform this task in less than a second. However, if the key length is extended to 32, it would need 65,536 combinations to be tried, needing a few seconds. Extending this, the number of combinations for a 256-bit key would need literally many years for even a powerful computer to crack.

While the key length is an important factor, the mathematical algorithm used for processing encryption and decryption is also equally important. The algorithm is supposed to quickly perform the action, while maintaining necessary data and key security. There are many algorithms, such as SHA1, 3DES, etc. There are two types of keys symmetric and asymmetric. The symmetric types use only one key for encryption and decryption, while for asymmetric keys, there are two different keys, which are complementary to each other. Please refer to Figure 1, which shows the basic cryptography functionality, which is designed with the objectives of data confidentiality, integrity and authentication in mind.

It is important to understand what cryptography means to the Internet. The Internet is blessed with SSL (Secure Socket Layer) and TLS (Transport Layer Security) protocols, which perform the job of encrypting and decrypting the data sent, so as to enable users to securely exchange personal information, credit card numbers, etc. SSL and TLS are based on asymmetric keys exchange. There are two demands here that the data must be encrypted for security reasons, and that the website must be known to be legitimate. The latter is important because attackers may host websites to steal personal information. To ensure legitimacy, the Web server has an SSL certificate, which enables traffic via the HTTPS protocol, using TCP port 443 for communication. This SSL certificate is provided or signed by a trusted certification authority such as Verisign or Thawte, which ensures that the SSL certificate holder is a genuine party, will adhere to security standards, and is eligible to obtain a certificate and install it on their servers. The SSL certificate is tied to the domain name, such as abcd.com.

To understand all the security concerns, we need to first study how certificates work. Digital certificates using asymmetric PKI have two keys, a public key and a private key. The private key is installed on the Web server where the website URL is supposed to be secured using SSL. The public key is shipped with all browsers that support SSL. To support multiple certificate authority vendors, browsers are equipped with their public keys, as well as various ciphers (the encryption and decryption algorithms). Each public key has an expiration date and needs to be updated once it nears expiration. When we install a digital certificate on a Web server, we are essentially installing the private key, which is specifically created by the trusted certification-providing authority.

Now let’s see how this mechanism works at a higher level, for a browser. When a person tries to access an SSL-secured website, the browser first challenges the server by sending its own cipher strength. In response, the server does the same, and also sends a copy of its own installed SSL certificate (for the hosted website URL). At this point, the browser checks the validity and authenticity of the certificate,

by using the set of public keys on it. On finding it to be acceptable, the browser sends back a digitally signed response to the server to initiate further secure communication. If the server certificate cannot be verified for authenticity, the browser alerts the user about this situation. It’s important to note that while SSL helps achieve security, there is an overhead, since TCP/IP by default does not provide any security adding the encryption layer onto the existing protocol frame can result in bigger TCP packets.

Cryptographic attacks

Network administrators commonly invest time and money to design security around applications, servers and other infrastructure components, but tend to take cryptographic security less seriously. Before going into the various attacks, let’s understand first that cryptography is all about keys, the data, and the encryption/decryption of the data, using the keys. A few cryptographic attacks try to decipher the key, while others try to steal data on the wire by performing some advanced decryption. Let’s take a look at a few common attacks on cryptography.

The SSL MITM attack: In this case, the attackers intrude into the network and establish a successful man-in-the-middle connection. The attackers silently watch the HTTPS traffic on the wire, and wait for the targeted website to respond to some browsers HTTPS request. As we learnt earlier, the server is supposed to send its digital certificate to the browser as a part of the SSL handshake process. The attackers grab this certificate, and note down various details such as the domain name, expiration date, cipher strength, etc. The attackers then create their own certificate (also called a self-signed certificate), containing the same information as that of the captured certificate. From this point on, the man-in-the-middle attackers intercept each browser request and respond with the fake certificate. As a normal response to such a situation, the Web browser pops up a warning to the user, which in most cases is ignored, and thus the attackers are successful. Further, on the server side, the attackers establish a separate HTTPS connection to complete the request, and the result of the response is fed back into the browser on the

connection already established. This gives the attackers complete control on the SSL traffic, and helps them steal the personal information. Since this attack involves a real intrusion into the network, it is less likely to happen, but can result in serious data loss. Also, since the attackers are not breaking the request-and-response chain, it becomes tough to detect the data theft.

The SSL MITB attack: Similar to the attack mentioned above, in this case, the attackers inject a JavaScript code snippet into the browser to create a man-in-the-browser situation. This snippet monitors all SSL activities and records the session. While this is happening, the attackers also record the encrypted version of the same session, and programmatically try to find out the cipher strength and the key, besides stealing data. This attack is becoming more popular of late, due to multiple open source browsers, and the various security vulnerability problems with each of them.

Key hijacking: This is another intrusive type of attack, whereby the attackers gain access to the Web server hosting the website (by using one of the many intrusion techniques already discussed in previous articles of this series). Once the server is compromised, the attackers use an elevated privilege attack to gain access to the certificate store, from where the private key can be obtained. The attackers then use packet sniffing to download an entire HTTPS session, and store it for offline decryption. The decryption process needs the private key, which is already stolen; and the public key, which is available in the browser's trusted authority key store. The data set so deciphered might reveal vital personal information such as user IDs, addresses, credit-card numbers, etc, assuming that the targeted website sells goods online using e-commerce technology.

The birthday SSL attack: This attack relies on a mathematical theory called the birthday problem, which says that statistically, in a set of randomly selected people, some pairs of people will have the same birthday. The probability increases as the number of people grows. In cryptography, the data integrity is established using a hash or checksum, which is calculated at both ends of the transmission, to ensure that the data is not tampered with. Birthday attacks target the hash, and need

multiple attackers coming together, who individually capture chunks of data and share it among themselves. Each chunk is then programmatically analysed to create an additional set of data in such a way that the hash of it matches that of the data chunk. In other words, for a given chunk of data and hash combination, the mathematical algorithm creates a clone data set. Further processing of the original data chunk and the resultant data set helps derive the encryption key. This attack is very time-consuming and technically complex, but can be accomplished by using multiple powerful computing machines and software programs.

Chosen dataset attacks: As discussed earlier, attackers always aim for data as well as the key, in order to completely compromise a cryptographic system. A chosen dataset method consists of two different types. In the first case, called chosen plaintext, it is assumed that the attackers have access to the original data and the encrypted version of it. The attackers then apply multiple encryption keys to the original data, and each time the output is compared with the already encrypted version. If the result is positive, it means the key is derived. In the second type of attack, called chosen cipher text, attackers have the cipher text and also the decrypted version of it. Again, the attackers try multiple keys until the output matches that of the decrypted version obtained already. These attacks are a bit less time-consuming, but need the attacker to gain an enormous amount of data and computational power for the desired results.

The SSL brute-force attack: Here, the attackers send very small data sets to be encrypted by the SSL protocol. The attackers capture the result and store it against the transmitted dataset. By doing this for lots of data chunks, a key can be eventually derived. This process is very slow, and can take days to decipher the key, and it has been found that such attacks often originate from within the firm’s network. To speed up the process, this method is usually combined with the group key decipher attack.

Group key deciphering: As learnt earlier, key-based encryption is dependent on the length of the key; a bigger key requires more time to decipher. In a group key

deciphering attack, multiple attackers come together each one with a powerful machine. Unlike brute force attacks, where a lot of data is captured, in the group method only a given set of data is captured and used. This data is subjected to all possible permutations of keys to try decrypting the data. Since 256-bit encryption can take many years to decipher, using multiple powerful computing machines can bring that time down. Attackers also use statistical grouping of keys to be tried from different machines, to further quicken the process. In the past, a few such experiments showed that cracking a 128-bit key required only a few days. With improving CPU speeds and throughputs, unfortunately, cracking a 1024-bit key quickly can become a reality soon.

Compromised key attacks: Cryptography is all about trust, whereby a trusted certificate-providing authority signs a certificate. The provider itself is supposed to be extremely secure. Unfortunately, in the past, the certificate-providers own private key has either been exposed or stolen by attackers, who then have used this private key to sign certificates created for a domain name, which is their own site. Any browser lured to this website will not suspect such a website, because the certificate will pass the authenticity test. This happens because the public key of such certificates will already be present in the browser certificate store. This can, and in the past has, resulted in loss of personal information.

SSL DoS attacks: An attacker’s main aim is usually to steal data. Since it is a troublesome and highly technical process in cryptography, a few attackers tend to use legacy methods, such as denial of service attacks. SSL negotiation adds an overhead to the TCP protocol, slowing down the communication to achieve security. In an SSL denial of service attack, the attackers establish SSL communication through a browser, and then send multiple bogus packets with varying lengths on that channel. Each packet is decrypted and processed on the server side, thus eventually exhausting CPU power, resulting in service outage. In another form, which takes place at OSI Layer 3, TCP port 443 is bombarded with bogus fragmented packets, creating a similar effect.

Protecting FOSS systems

In the FOSS world, cryptography is mainly used on Web servers by implementing the SSL protocol. Besides, open source developers can digitally sign the code before sending it to a trusted party, to prevent wire-tapping. On a Web server, the very first step is to use a digital certificate from a trusted authority. It should also have the latest and strongest cipher algorithm, and the key length should at least be 256 bits. The second step is to protect the certificate store that crucial area on the Web server where the websites private key is stored. Only administrators and network managers should have access to it. To protect FOSS networks from brute-force attacks, other network security protection should be in place (this has already been discussed in previous articles in this series). While most critical infrastructures implement a firewall, a UTM device and powerful antivirus or anti-Trojan software, it is also imperative to have an intrusion detection system (IDS) in place. IDS systems are capable of intercepting denial of service and brute-force attacks, and also help stop other critical anomalies. In case of Linux workstations, cryptography can be used to encrypt a file or entire disk too.

Source : http://www.opensourceforu.com/2013/05/cyber-attacks-explained-cryptographic-attacks/