Data-transmission Encryption Technology

Strengths and weaknesses in some of the most widely used alternatives

Victor Rudolfsson
Information Security
Høgskolen i Gjøvik
Gjøvik, Norway

I. INTRODUCTION

Today the safekeeping of data is often attributed as a responsibility of few rather than a responsibility of us all. Many of us expect our data to be safe when we give it to a party requesting it, such as a bank, a social networking website, or any other organization to which we may be required to provide data in exchange for its service. Few do however see this as their own responsibility to the extent that it may be, and even if data is perceived secure in storage, the majority of users do not see the need to secure data in transmission unless this is required by the requesting party.

This paper will seek to see how secure data really is unless an encryption method is used for transmission, and what can be done about it. What methods of securing data-transmission are most commonly used, and what are their strengths and weaknesses?

II. TECHNIQUES

There are many different techniques that can be used to secure data transmission, and more often than not different alternatives to each technique, but most of them build upon the idea of encryption. I'm going to analyze a few different available protocols used for this purpose, to find out how much difference they really make, and what one has to lose by utilizing them.

The most commonly used methods are VPNs, *Virtual Private Networks*, which is a concept with multiple protocol alternatives; TOR, or *The Onion Router*, which is a network relying on a big amount of computers through which data will be passing encrypted before reaching its destination; and SSL, *Secure Socket Layer,* which is an additional level of encryption using certificate authentication to provide a more secure connection between a client and a server.

All of these alternatives rely on the way we transmit data over the internet today, with a suite of protocols called TCP/IP [1][2]. TCP/IP contains four layers[3], and every protocol used with TCP/IP operates within one of these layers. Each of the four layers has its own responsibility, and it is in the second layer from the top called the *transport layer* that encryption protocols such as SSL (and it's successor TLS) are supposed to work with by providing asymmetric encryption from the application layer to the transport layer.

Because it is the responsibility of encryption protocols to provide encryption, this means that if no encryption protocol is used as by default, no encryption will be provided. This means that messages sent over the internet are sent in *clear-text* by default, and can be easily picked up and read.

A. *Virtual Private Networks*

VPN stands for *Virtual Private Network*, and as the name implies it establishes a private network between clients and the server, which allows the server to act as a gateway through which all data to and from the clients will pass, much like your modem and router would normally act on a local network – but the connection between the client and the server usually occurs over the internet, using an encrypted tunnel.

This means that for a website or a server a client is connected to, the traffic will seem to originate from the VPN server, which acts as an intermediate between the source and the destination, provided a layer of anonymity and security at the same time, as the source is never visible to the destination server, and the data sent between the client and the VPN server is encrypted.

There are multiple different protocols available for VPN connections – such as IPSec, PPTP, and Open VPN to name a few – but Open VPN is the one I will focus on in this paper.

The way Open VPN works is by creating an encrypted tunnel (usually a UDP tunnel, but it can also run over TCP). Using OpenVPN, a stream of encrypted UDP packets are sent between the client and server containing the data, using OpenSSL for encryption and relying either on certificates through a custom SSL/TLS implementation, or a pre-shared static key; meaning that data transmitted when using OpenVPN is encapsulated and encrypted inside a tunnel of packets until it reaches the VPN server. [7]

B. *The Onion Router (TOR)*

'*The Onion Router*', or TOR for short, is a distributed overlay network which aims to anonymize and encrypt online traffic [4]. It does this by layering every packet in multiple layers of encryption, and then selecting a random route through its network of relays through which data will pass between the source and destination. The *onion* in its name is a reference to the way data is encrypted, which is much like an onion with the data in the middle surrounded by several layers of encryption.

The encryption TOR uses is based on a stream cipher – 128-bit AES in counter-mode, the Diffie-Hellman protocol, a public key cipher based on 1024-bit RSA, and a hash function: SHA1. [8]

The first thing that happens when data is sent over TOR is that a random path is chosen, and the data will be layered in the same amount of layers of encryption as the amount of nodes in this path. Each node in the path then decrypts the top layer, until it reaches the final node in the path called the *exit node*, where the last layer of encryption is decrypted, and it then reaches it destination. Each node in the path only knows the previous node from which the data came, and which node is the next one in its path, but no node knows the whole path.

C. SecureSocketsLayer/TransportLayerSecurity

Because establishing host to host communication is one of the things that the transport layer is responsible for, it's important to mention that this is where encryption protocols such as SSL/TLS resides: Because if the security checks are not passed, connection will not be established between the hosts.

The SSL/TLS procedure consists of two steps, the first one being authentication and the second one being encryption. Because SSL/TLS is based on the idea of using certificates, both sides have to prove their identity to one another by showing their digitally signed certificates and validating these with the certificate authority before the authentication process is complete. Because SSL/TLS resides in the *transport protocol,* which is responsible for establishing the connection, a connection will not be established if the security checks are not passed.

During the next phase, this protocol makes use of two keys – a public and a private one. The public key is used to encrypt, whereas the private key is used to decrypt. Both clients share their public key with one another, and use their private keys to decrypt the data received once transmission has begun. In contrast to SSL/TLS, regular SSL only requires one side (the server) to verify its identity.

Because SSL/TLS does not require data to take a different route between the source destination, there is usually no noticeable speed difference, but it is up to the server if it allows SSL connections.

III. Conclusion

TOR has a rather obvious weakness: as each node decrypts one layer, the final node decrypts the final layer before it reaches its destination, which means that unless an end-to-end encryption such as TLS or SSL/TLS is used, traffic will be unencrypted at the final step.

As such, wiretapping the *exit node* may be the only feasible way to monitor traffic passing through the TOR network.

SSL has one of its main weaknesses in the user – because if a certificate with mismatching keys is presented from the server, most browsers will warn the user about this, but this is commonly disregarded by the user since many servers use outdated certificates, which can result in the same kind of warning, and thus the chance that the user will accept a mismatching certificate is fairly high.

This means that if a third party would act as a kind of invisible proxy in between the server and the client, and provide the user with a false certificate pretending to be the server, the user may disregard this as an outdated certificate and accept it regardless – ultimately accepting the third party as a relay between the user and the server.

However, SSL is made to be resistant to brute-force attacks (although one should note that it's not impossible, successful attempts have been made), and to make it very hard to decrypt without access to the proper keys. TOR, on the other hand, is resistant to brute force attacks because of its multiple layers of encryption and its implementation of several different components to apply it. Both SSL and TOR have weaknesses that lie outside the scope of what can be done without directly attacking either client, or server.

Finally, even though SSL and SSL/TLS is an encryption protocol for data-transmission, it can be regarded as more of a complement to any existing connection – whether encrypted or not. Regardless of whether a VPN or TOR is already in use, SSL can be used as a complement to data is encrypted end-to-end, and the biggest weakness with SSL and SSL/TLS is in the end to not use it where available.

References

[1] Structure of a TCP packet, RFC 793: Transmission control protocol, September 1981, section 3.1 page 15, (http://tools.ietf.org/html/rfc793),

[2] Structure of UDP packet, RFC 768: User Datagram Protocol, 28 August 1980, http://tools.ietf.org/html/rfc768

[3] "*Layers of TCP/IP",* The TCP/IP model http://technet.microsoft.com/en-us/library/cc786900(v=ws.10).aspx

[4] "*The Onion Router*", www.torproject.org

[5] "*What is TLS/SSL?*", http://technet.microsoft.com/en-us/library/cc784450(WS.10).aspx

[6] "A*uthentication and data exchange",* http://technet.microsoft.com/en-us/library/cc783349(v=ws.10).aspx#w2k3tr_schan_h ow_hkrr

[7] "*OpenVPN cryptographic layer*", http://openvpn.net/index.php/open-source/documentation/security-overview.html

[8] TOR Protocol specification, https://gitweb.torproject.org/torspec.git/blob/HEAD:/tor-spec.txt

[9] "*Rogue Nodes Turn Tor Anonymizer Into Eavesdropper's Paradise*", Wired Magazine, 2007-09-10