International Journal of Computer Applications (0975 – 8887)
Volume 175 – No.5, October 2017

Secure Client - Server Communication through SSL

Lalit Mohan Joshi
PhD Research Scholar
B.T.K.I.T. Dwarahat (India)

ABSTRACT

Computer security has become one of the most important concerns in the entire discipline of computing. The recent explosive growth of the Internet and the World Wide Web has brought with it a need to protect sensitive communications over the open networks. In the past, security violations were generally done by Young adults, just for fun. But as technology and usage of internet increased, there is always the threat of planned attack (cyber terrorists), where the loss of money could be large in billions, Hence the need for secure connection will arise. A Robust solution for this is provided by VPN (Virtual Private Networks) and SSL (Secure Socket Layer) protocols in my research .In the recent past SSL protocol has revolutionized the area of VPN (Virtual Private Network). To set up secure communication from virtually any Internet-connected web browser, SSL based VPN products permit users to do such a thing. To implement a secure remote access, It is easier and resourceful than its predecessor (IP Sec).

In my research paper in which security of client – server communication is achieved by principles of security, like Authentication and Encryption. There are two side of this application server side and client side. Without certification, client can't communicate with the server. The SSL handshake-protocol message flow involves client and server negotiating a common cipher suite acceptable to both parties. The application based on RSA algorithm, using for encryption especially for data sent to server. A certificate is issued to each server and client using key tool commands. Private keys are protected by a password in key store. Client and server can thus communicate with each other only if the certificate has been issued to both. Once issued a secure and safe communication link is established. It gives an insight into the different attacks on the internet, and how the data can be sent securely through SSL.

Keywords

Secure Socket Layer, Encryption, Security.

1. INTRODUCTION

In this we would introduce you the topic of VPN 'Virtual Private Network', the back bone of this project. This gave us motivation regarding secure remote access, to learn it, deploy and find new implementations. A private communications network frequently used within companies, organizations, institutes is called Virtual Private Network; to communicate over a public network. VPN has drawn the attention of many organizations to inflate their networking possibilities as well as moderate their costs. The idea behind with the VPN is called Tunneling. VPN tunneling consist of to set up and uphold a logical network connection (that may involve with intermediate hops). On this connection, packets are encapsulated within carrier protocol or some other base. These packets are constructed in a specific VPN protocol format then in between VPN client and server they are transmitted, and at last on the receiving side, they are de-encapsulated. Each packet is encapsulated can provide: **Confidentiality, Integrity, Authenticity, Non-repudiation.** Obviously these are the four basic properties of Information Security. Ex:-Confidentiality is the extreme essential security property in military surrounding as well as in the Bank. There is also a most significant feature of security is called Integrity of the data. In the path of communication data has not been modified, this is confined by the Integrity. Authenticating is just confirming that the sender is reliable and trust worthy. Here non-repudiation means that the sender and the recipient were present, in fact sending or receiving the message are claimed by the parties, respectively. In short, the origin of non-repudiation verifies that data has been sent, on the other hand delivery side of non-repudiation justify its been received.

1.1. Classification

The Virtual private networks can be classified into two main categories as follows: Secure and Trusted.

1.1.1. Secure VPN

It uses cryptographic tunneling protocols to provide the necessary confidentiality (preventing snooping), sender authentication (preventing identity spoofing), and message integrity (preventing message alteration) to achieve the privacy intended. When properly chosen, implemented, and used, such techniques can provide secure communications over unsecured networks. Because such selection, implementation, and use are not irrelevant, Now a days on the market there are many insecure VPN schemes. If we talk about security then, Secure VPN protocols have following features:

- IP Sec (IP security), IPv6 compulsory part
- SSL used to tunnel the huge network stack, such as in Open VPN, as well as to secure from what is essentially a web proxy.
- PPTP (point-to-point tunneling protocol), jointly generated by a number of companies, indulging Microsoft.

For business customers those want the security and convenience of a VPN, some vast ISPs offer "Managed VPN service", in one condition that do not undertake administration of VPN server by themselves. Additionally with secure access to their employer's internal network is to provide remote workers. Sometimes as a part of the package other security and management services are included. These services are to keep and update the anti-virus and anti-spyware programs on each and every computer of clients.

1.1.2. Trusted VPNs

It does not use cryptographic tunneling, and instead rely on the security of a single provider's network to protect the traffic. Multi-protocol label switching (MPLS) is commonly used to build trusted VPNs. Other protocols for trusted VPNs include:

* L2F (Layer 2 Forwarding), developed by Cisco.

* L2TP (Layer 2 Tunneling Protocol), including work by both Microsoft and Cisco.
* L2TPv3 (Layer 2 Tunneling Protocol version 3).

1.2 VPN Architectures

Intranet VPN is used to make connection among fixed locations such as branch offices. This kind of LAN-to-LAN VPN connection joins multiple remote locations into a single private network

Extranet VPN is used to connect business partners such as suppliers and customers. This kind of VPN allows various parties to work in a shared environment.

Remote Access VPN is a user-to-network connection for the home user and mobile user connecting to corporate private network from various remote locations.

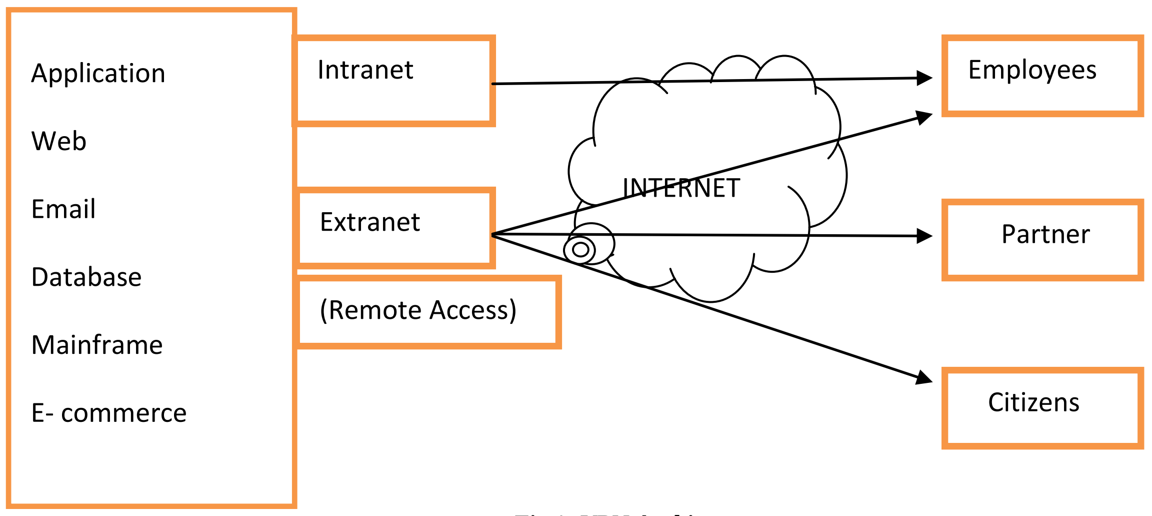


Fig 1: VPN Architecture

1.3 Typical Elements of a VPN connection:

1.3.1 VPN server

A computer accepts VPN connections from VPN clients. A VPN server can provide a remote access VPN connection or a gateway-to-gateway VPN connection.

1.3.2 VPN client

A computer initiates a VPN connection to a VPN server. A VPN client can be a remote computer obtaining a remote access VPN connection or a router obtaining a gateway-to-gateway VPN connection.

1.3.3 VPN tunnel

The portion of the connection in which data is encapsulated and encrypted.

1.3.4 Tunneling protocols

The communication standards used to manage tunnels and encapsulate data.

1.3.5 Tunneled data

Data that is encapsulated and encrypted, and sent across a private link.

1.3.6 Transit network

The shared or public network such as a private intranet or the Internet where the encapsulated data pass through it.

1.4 Advantages Of VPN

Listed below are some benefits provided by VPN:

1.4.1 Extend geographic connectivity

VPNs employ the Internet for inter-connectivity between remote parts of an intranet. Because the Internet is accessible globally, even the most far flung branch offices, users, and mobile users (such as salesmen) can easily connect to the corporate intranet.

1.4.2 Improve security for remote user and network connection

Because VPNs use the tunneling technology to transmit data across "unsecured" public networks; data transactions are secure to an extent. In addition to the tunneling technology, VPNs use extensive security measures, such as encryption, authentication, and authorization to ensure the safety, confidentiality, and integrity of the data transmitted. As a result, VPNs offer a considerably high degree of transaction security.

1.4.3 Reduce Implementation and operational costs

VPNs cost considerably less than the traditional solutions, which are based on leased lines, Frame Relay, ATM, or ISDN. This is because VPN eliminate the need for long-distance connections by replacing them with local connections to a carrier network. The organization does not need to employ as many trained and expensive networking personnel as it would if the VPN were managed by the organization itself.

1.5 OSI Network Model

In 1984, Open Systems Interconnect, a group dedicated to providing international standards, released an abstract frame work for classifying technologies involved in inter-computer communications. It consists of seven layers (or levels) and their corresponding VPN technologies are shown in the following figure.

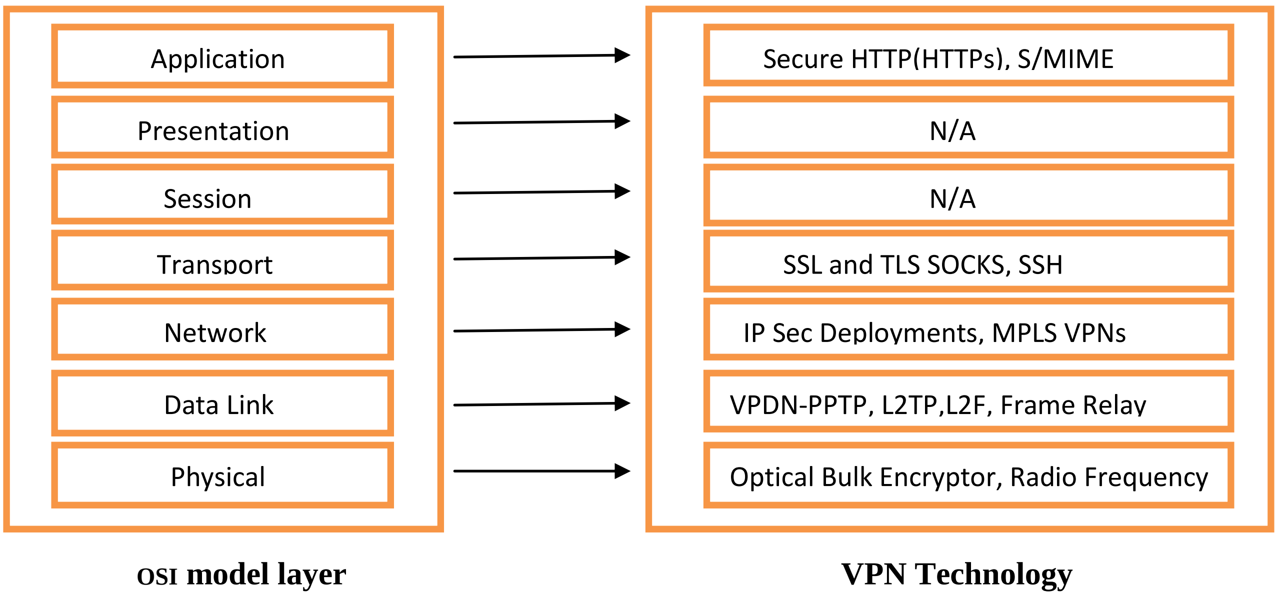

Application
Presentation
Session
Transport
Network
Data Link
Physical
Secure HTTP(HTTPs), S/MIME
N/A
N/A
SSL and TLS SOCKS, SSH
IP Sec Deployments, MPLS VPNs
VPDN-PPTP, L2TP,L2F, Frame Relay
Optical Bulk Encryptor, Radio Frequency
OSI model layer
VPN Technology

Fig 2: VPN Architecture with OSI model layer

Historically, VPN tunneling was typically performed at the Network Layer or lower (e.g., IPSec, which operates at the Network Layer).SSL VPNs work differently. They establish connectivity using SSL, which functions at Levels 4-5. They also encapsulate information at Levels 6-7 and communicate at the highest levels in the OSI model. Today, some SSL VPNs are also able to tunnel network level information over SSL, making SSL the most versatile remote access VPN technology available.

1.6 Important Security protocols

There are two main trusted and widely used protocols in the implementation of these VPNs. IPSec and SSL, as mentioned above both come into category of Secure VPNs. Algorithms for best encryption and more discursive authentication as a superior security features provided by **"Internet Protocol Security"** Protocol (IPSec). IPSec uses three main protocols.

1.6.1 IP Authentication Header (AH), IP protocol 51

It provides authentication of the origin of the data, ensures data integrity, and protects against replay.

1.6.2 IP Encapsulating Security Payload (ESP), IP protocol 50

It protects data from viewing by third parties, and provides the same features as AH.

1.6.3 The Internet Security Association and Key Management Protocol (ISAKMP)

Payload formats defined by this protocol framework. This framework of IP also describe the mechanics to implement a key Exchange protocol, and security association negotiation.

IPSec can be used in two modes: In which Tunnel encrypts the header and the each packet payload while transport only encrypts the payload.

- In between the end points of a connection Transport mode is generally used - for example, to set up a secure connection like a VPN between a server and a client device.
- In between two systems where one or both of them are not the end point of the connection, here the tunnel mode is normally used. For example, to communicate between a firewall and a VPN server on a LAN, or between a remote dial-in system and an edge device such as a router or gateway, you might use tunnel mode (or IPSec tunneling).

2. SECURE SOCKET LAYER

Secure Sockets Layer" protocol is the standard security technology for establishing an encrypted link between a web server and a browser. This was developed by Netscape in 1994, which allows clients (Web browsers, typically) and HTTP servers to communicate over a secure connection. The huge amount of data which is passed between the web server and browsers remain private and integral, confirmed by thi link. An industry standard, called SSL used by most of the websites to protect their online transactions of their customers. To set up a secure remote access sessions from virtually any Internet-connected web browser is allowed by SSL VPN product.

There are numerous variants of SSL: In 1994, SSL 1.0 was introduced in the Mosaic browser, Because of the shortcomings in security, SSL 2.0 is hardly used today; On the other apart of SSL, The SSL 3.0 is universally supported; and finally the Transport Layer Security(TLS), which is an improvement on SSL 3.0, has been adopted as an Internet standard and is supported by almost all recent software.

In 1996, as the relevance of the World-Wide Web as a respectable mainstream business channel And communications mechanism became assured, and as SSL emerged as an effectively unchallenged de facto standard, the Internet Engineering Task Force (IETF) began formal standardization of the SSL protocol. In 1999, it completed its work and established SSL as the official standard for secure Web communications under the name Transport Layer Security (TLS).

SSL/TLS is the most widely deployed security protocol in the world [Res01]. As such, it has under gone extensive scrutiny and has yet to be degraded by any known weakness. This does not mean it is guaranteed secure for the future, but it does mean that many of the brightest minds in cryptography and mathematics have been unable to find any holes in its cryptographic armor. In the past, SSL/TLS was a general protocol that would be tightly coupled with specific applications, thus the extreme confusion about what an SSL VPN really is. It would be used to secure session communication between two hosts using a single application or protocol at atime. The most well-known use of SSL is in the HTTPS protocol to enable secure web-based ecommerce. SSL is the default security solution for application to application needs, but it has never been implemented to handle arbitrary multiple protocols at the same time, until now.

SSL encrypted pages had the prefix HTTPS whereas non-encrypted pages used a prefix of HTTP. Today in addition to the change in the URL, most browsers display some icon indicating that the session is encrypted—a lock, a key, etc. HTTPS is the web protocol that utilizes SSL to encrypt HTTP, and is used worldwide today for secure web communications. HTTPS should not be confused with S-HTTP, an alternative method for encrypting web communications. S-HTTP is designed to send individual messages securely whereas SSL is designed to establish a secure connection between two computers. S-HTTP competed with HTTPS in the early days of secure web sessions, but was generally considered inferior to HTTPS for e-commerce use (as it did not encrypt some session information), and is no longer used. S-HTTP is an extension to the HTTP protocol used to support sending data securely over the World Wide Web.

Today, servers typically provide regular web service HTTP on port 80, and SSL encrypted web traffic HTTPS over port 443. Overleaf is an HTTPS encrypted web page. Note the https at the start of the URL and the lock icon near the bottom right of the screenshot.

2.1 SSL in TCP/IP Protocol Stack

As the name Secure Sockets Layer indicates, SSL connections act like sockets connected by TCP. Therefore, you can think of SSL connections as secure TCP connections since the place for SSL in the protocol stack is right above TCP and below the application layer as shown in figure. It is important to note, however, that SSL doesn't support some of the TCP features such as out-of-band data.

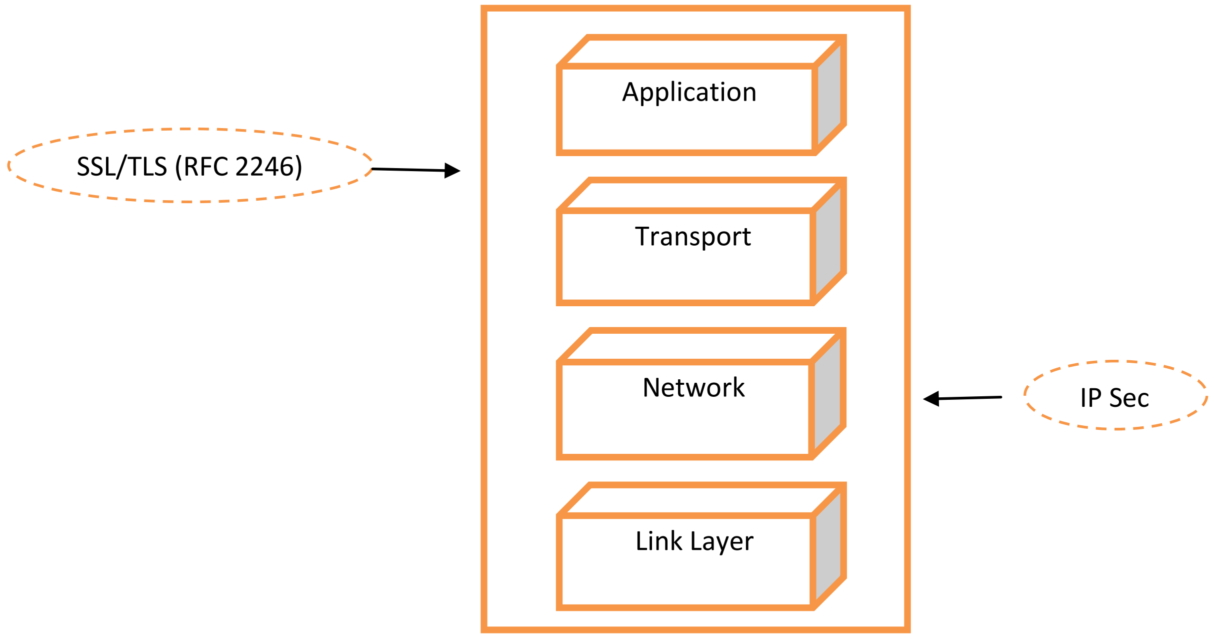


Fig. 3: SSL vs. IP Sec

Among the features of SSL that have made it the de facto standard vehicle for secure ecommerce transactions is its support for negotiable encryption and authentication algorithms. The designers of SSL realized that not all parties will use the same client software and consequently not all clients will include any particular encryption algorithm. The same is true for servers. The client and server at the two ends of a connection negotiate the encryption and decryption algorithms (cipher suites) during their initial handshake. It may turn out that they do not have sufficient algorithms in common, in which case the connection attempt will fail. Security's worst enemy is complexity and SSL VPN defeats this enemy. Note that while SSL allows both the client and the server to authenticate each other, typically only the server is authenticated in the SSL layer. Clients are customarily authenticated in the application layer, through the use of passwords sent over an SSL-protected channel. This pattern is common in banking, stock trading, and other secure Web applications.

Compared to IPSec VPNs, SSL VPNs provide:

- Clientless Access,
- Compatibility with any Remote Access situation,
- More Security, with granular access control,
- More control for IT,
- Easier management, with lower total cost of ownership.

· Provide faster ROI (return on investment) than other protocols: Return of Investment is your annual benefit divided by your investment amount. By replacing your IPSec VPN Transport Network IPsec Over Network Layer SSL/TLS (RFC 2246)Between Application and Transport layer10with SSL VPN, you will see bottom line benefits in the areas of operations, security productivity, and business partner collaboration. Up gradation of only1000 IPSec VPN users to SSL shown by the statistics, you could assume operational assistance of $115,200 and work rate benefits of $3,600,000 over 3 years, over an extranet to extend it to 1000 of your business members, you could also add additional cooperative benefits of $300,000,resulting in summative benefits of $4,015,200. It very easier, cheaper, and faster to deploy than IPSec VPNs with the solutions of SSL VPN which has no client, because there is no difficult-to-configure, difficult-to-manage, and difficult-to-support components.

2.2 Establishing Secure Tunnels Using SSL:

Now that we understand what SSL is and how it works, let us discuss how SSL allows us to create tunnels. A secure tunnel between computers can best be understood as a secure channel of communication between two machines that have an in secure environment between them. The communications tunnel is like a tunnel under a river that allows automobiles to travel from point A on one side of the river to point B on the other side and prevents environmental elements like water from interfering with the traffic as shown in the diagram below. Secure Communications between two computers over public network allowed by communication tunnel, benefit of this tunnel is that it does not allow other computer to access the shared data between two machines on those networks.

However, unlike the example of the tunnel under the river, network tunneling does not employ a physical barrier between the two computers communicating and other machines. Rather, encryption of all sharing media in between two computer involved by the related tunneling, in case even if another computer were to receive the data from sharing media, it would be unable to decipher the actual message’s contents, between the machines. To achieve or implement its goals secure tunnels created by SSL VPNs

2.2 Basics of Tunneling Technology

As I described about tunneling, this lies at the core of VPN technology. While the cost performance of VPNs is attributed to the use Internet use, VPNs offer are the direct result of tunneling which comes under the category of safety and security. In this chapter, you'll learn about the basics of tunneling technology and the role it plays to ensure the safety of transactions across a widely heterogeneous internetwork, such as the Internet. You'll learn about the components of tunneling, the working of tunneling technology, and the format of a tunneled packet. You'll also learn about the different types of tunnels and tunneling protocols.

Tunneling is the most significant component of VPN technology. It allows organizations to create virtual networks across the Internet and other public networks. This virtual network cannot be accessed by "outsiders"—users or computers that are not a part of the organization's intranet. Tunneling is the technique of encapsulating an entire data packet in the packet of another protocol format. In other words, the header of the tunneling protocol is appended to original packet. The resultant packet is then transferred to the destination node or network across the intermediate infrastructure .The most important aspect of tunneling is that the original data packet, also referred to as the payload can belong to an unsupported protocol. Instead of transferring the original packet, which might not be routable across the intermediate infrastructure, the underlying tunneling protocol appends its header to the tunneled packet. This header provides the requisite routing information so that the packet can be successfully delivered across the internet work.

Tunneling is analogous to sending a letter. After you write a letter, you place it in an envelope. This envelope displays the address of the recipient. When you post this letter, it is delivered to the recipient according to the address on the envelope. The recipient then needs to open the envelope to read the letter. In tunneling technology, the letter is equivalent to the original payload and the envelope represents the packet of the routable protocol in which the payload is encapsulated. The address on the envelope represents the routing information that is appended to the packet .When a tunneled packet is routed to the destination node, it travels across the internetwork through a logical path. This logical path is referred to as a tunnel. Upon receiving a tunneled packet, the recipient returns the packet to its original format.

2.4 Advantages of Tunneling

Tunneling offers a few advantages that have made a significant impact on networking technology. These advantages are listed below:

2.4.1 Simplicity and ease of implementation

Because the basic idea behind tunneling is simple, the technology itself is simple and easy to implement. Moreover, there is no need to change the existing infrastructure to accommodate tunneling technology, which makes tunneling a viable and lucrative solution for large as well as medium scale organizations.

2.4.2 Security

An organization's tunnel access is prohibited to unauthorized users. As a result, data traveling through the tunnels is relatively safe—despite the fact that the data is being transmitted across an unsafe and public medium, such as the Internet.

2.4.3 Cost-effectiveness

Tunneling uses public networks as the intermediate internet-works to transfer data to a destination. This makes tunneling an extremely cost-effective solution, especially when you compare it to the cost of implementing private intranets that span the globe or long-distance leased lines. In addition, an organization can save the considerable amount of money that it would have to spend annually on the administration and maintenance of these expensive solutions.

2.4.5 Protocol indifference

Data belonging to non-routable protocols, such as Network Basic Input/Output System (NetBIOS), and Net BIOS Enhanced User Interface(NetBEUI) is not compatible with the Internet protocols TCP and IP. Therefore, these data packets cannot be routed as-is across the Internet. However, tunneling allows you to route such non-IP packets successfully to the destination by "enveloping" them within IP packets.

2.4.6 IP address savings

As mentioned earlier, tunneling allows protocols with nonroutablenon-IP addresses to be inserted within a packet that uses a globally uniquepublic IP address. As a result, instead of having to buy and assign a globally unique IPaddress (also known as a public IP address) for each node within the network, the network can buy a small block of globally unique IP addresses. When a node within this private network establishes a VPN connection, any of the available IP addresses from the block can be appended to the non-IP data packets .Thus, the private network can reduce an organization's need for globally unique IP addresses.

2.5 Components of Tunneling

To successfully establish a tunnel between the two communicating ends, four tunneling components are required.

2.5.1 Target network

The network that contains the resources that needs to be accessed bythe remote client, which initiated the VPN session request. (The target network is also referred to as home network in some VPN-related documentations.)

2.5.2 Initiator node

The remote client or server that initiates the VPN session. The initiator node can be a part of a local network or can be a mobile user using a laptop.

2.5.3 HA (HomeAgent)

The software interface that generally resides at the network access node (router) in the target network. However, a destination node, such as a dial-up access server, can also host the HA. The HA receives and authenticates the incoming requests to verify that they are from trusted hosts. Upon successful initiator authentication, the HA allows the establishment of a tunnel.

2.5.4 FA (Foreign Agent)

The software interface that resides either at the initiator node or at the network access node (router) of the network to which the initiator node belongs. The initiator node uses the FA to request a VPN session from the HA at the target network .

2.6 Tunneled Packet Format

As described earlier, before it is delivered to the target network across the tunnel, the originaldata packet is encrypted by the FA. This encrypted packet is referred to as the tunneled packet.A tunneled packet consists of three parts. These parts follow:

2.6.1 Header of the routable protocol

This header field contains the addresses of the source(FA) and the destination (HA). Because transactions over the Internet are predominantly IP-based, this header is generally the standard IP header and contains the IP addresses of the FA and HA involved in the transaction.

2.6.2 Tunnel packet header

This header contains the following five fields

Protocol type indicates the type of the original data packet (or payload) protocol.

Checksum contains the checksum that is used to check whether the packet was corrupted during the transmission. This information is optional.

Key as an information is used to identify or authenticate the actual source of data (initiator).

Sequence number contains a number that indicates the sequence number of the packet in the series of packets that are being transmitted.

Source routing contains additional routing information. This field is optional.

Payload as a original packet sent by the initiator to the FA. It also contains the original header.

2.7 Tunneling Protocols

Tunneling technology makes use of three types of protocols.

2.7.1 Carrier protocol

These protocols are used to route tunneled packets to their intended destination across the internetwork. The tunneled packets are encapsulated within the packets of this protocol. Because it must route the packets across a heterogeneous internetwork, such as the Internet, this protocol must be widely supported. As a result, if tunnels are created across the Internet, the carrier protocol that is used predominantly is IP. However, in the case of private intranets, native routing protocols can also serve as carrier protocols.

2.7.2 Encapsulating protocol

These protocols are used to encapsulate the original payload. In addition, the encapsulating protocol is also responsible for the creation, maintenance, and termination of the tunnel. Today, PPTP, L2TP, and IPSec are the most commonly used encapsulating protocols.

2.7.3 Passenger protocol

The original data that needs to be encapsulated for the purpose of transmission across the tunnel belongs to this protocol. PPP and SLIP (Serial Line Internet Protocol) are commonly used passenger protocols.

2.8 Goals ofSSL

2.8.1 Encryption

It protects data from unauthorized use by converting it to an apparently meaningless form before transmission. The data is encrypted by one side (the client or the server), transmitted, decrypted by the other side and then processed.

2.8.2 Source Authentication

It is a method of verifying the data sender's identity. The first time a browser or other client attempts to communicate with a Web server over a secure connection, the server presents the client with a set of credentials in the form of a certificate. Certificates are issued and validated by trusted authorities known as certification authorities (CAs). A certificate represents the public-key identity of a person. It is a signed document that says: I certify that the public key in this document belongs to the entity named in this document. Note that the certificates used with SSL/TLS today are X.509 certificates.

2.8.3 Data integrity

It refers to means of ensuring that data has not been modified in transit. Besides protecting data through encryption, SSL uses hashing to ensure that the contents of communications session are not modified between the time one computer sends a message and the time the recipient reads it. It offers encryption, source authentication, and data integrity as means to protect information exchanged over insecure, public networks. Basic understanding of SSL will help us understand the workings of SSLVPNs, so let us look in to the Architecture of SSL protocol.

2.9 SSL Architecture

This consists of three major sub protocols namely: Handshake protocol, Record protocol, and Alert protocols.

Handshake as a mechanism is called Session creation, and allows client and server authentication.

Record protocol provides Confidentiality and Message Integrity.

Alert protocol uses to report errors.

The Change Cipher Spec sub-protocol is used to change the keying material used for encryption between the client and server. The key is computed from the information exchanged by the Handshake protocol. Before data can be sent across an SSL connection, the two ends must negotiate and exchange key information.

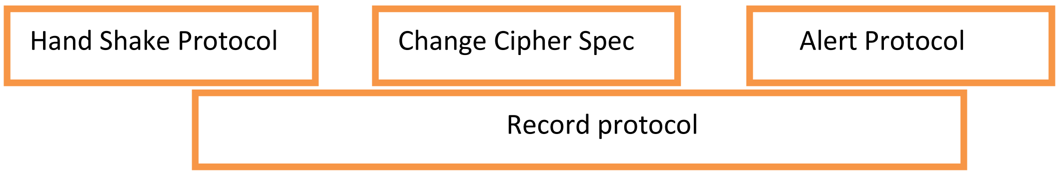


Fig. 4: SSL Architecture

2.10 SSL hand Shake messages

These messages individually explained as follows,

2.10.1 Client Hello

The client sends the server information such as SSL protocol version, session id, and cipher suites information such cryptographic algorithms and key sizes supported.

2.10.2 Server Hello

The server chooses the best cipher suite that both the client and server support and sends this information to the client.

2.10.3 Certificate

The server sends the client its certificate which contains the server's public key. While this message is optional, it is used when server authentication is required.

2.10.4 Certificate Request

This message is sent only if the server requires the client to authenticate itself. Most e-commerce applications do not require the client to authenticate itself.

2.10.5 Server Key Exchange

This message is sent if the certificate, which contains the server's public key, is not sufficient for key exchange.

2.10.6 Server Hello Done

This message informs the client that the server finished the initial negotiation process.

2.10.7 Certificate

This message is sent only if the server requested the client to authenticate itself.

2.10.8 Client Key Exchange

The client generates a secret key to be shared between the client and server.

2.10.9 Certificate Verify

If the server requested to authenticate the client, this message allows the server to complete the authentication process.

2.10.10 Change Cipher Spec

The client asks the server to change to encrypted mode.

2.10.11 Finished

The client tells the server it is ready for secure communication.

2.10.12 Change Cipher Spec

The server asks the client to change to encrypted mode.

2.10.13 Finished

The server tells the client it is ready for secure communication. This marks the end of the SSL handshake.

2.10.14 Encrypted Data

The client and server can now start exchanging encrypted messages over a secure communication channel.

2.10.15 Alert sub-protocol

It is used to report errors like:

- Unexpected message.
- Bad record MAC
- Decompression failure
- Handshake failure.
- Illegal parameters.
- To notify closure of connection.
- To notify absence of certificate.
- To notify that a bad or unknown certificate.
- To notify that a certificate is expired.

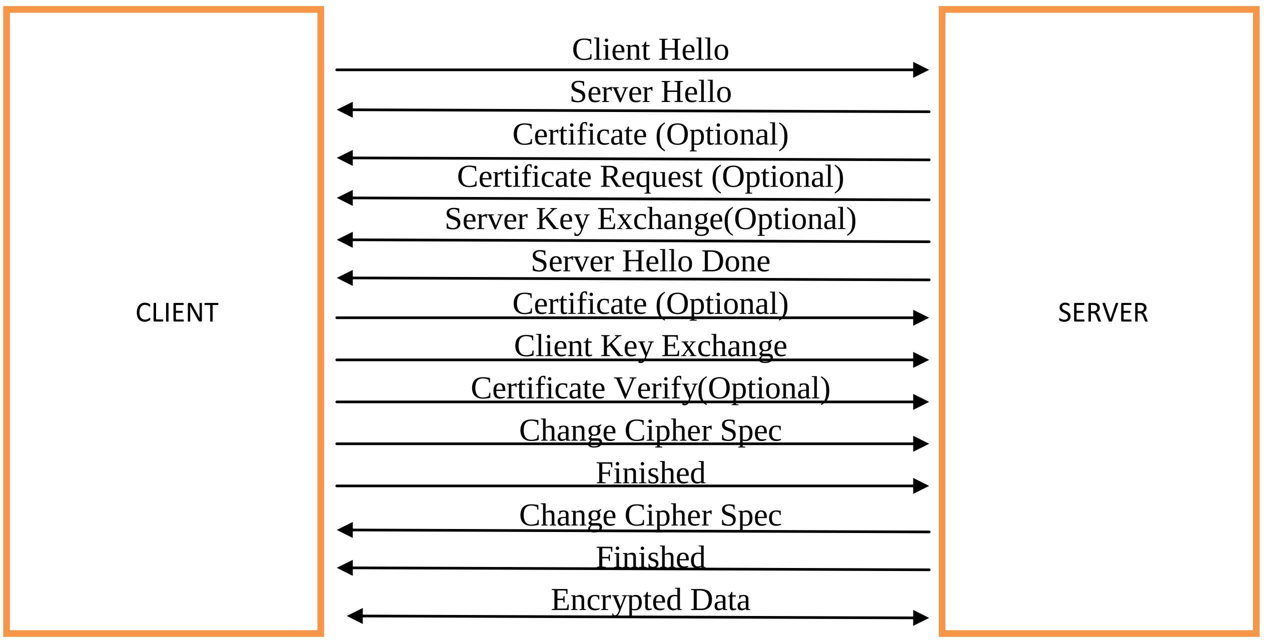


Fig. 5. SSL Messages

The Record sub-protocol provides Confidentiality and Message Integrity. This is explained in Fig 04**.** The Change Cipher Spec sub-protocol is used to change the keying material used for encryption between the client and server. It consists of a single message to tell other party in the SSL/TLS session, who is also known, is the peer that the sender wants to change to a new set of keys. The key is computed from the information exchanged by the Handshake protocol.

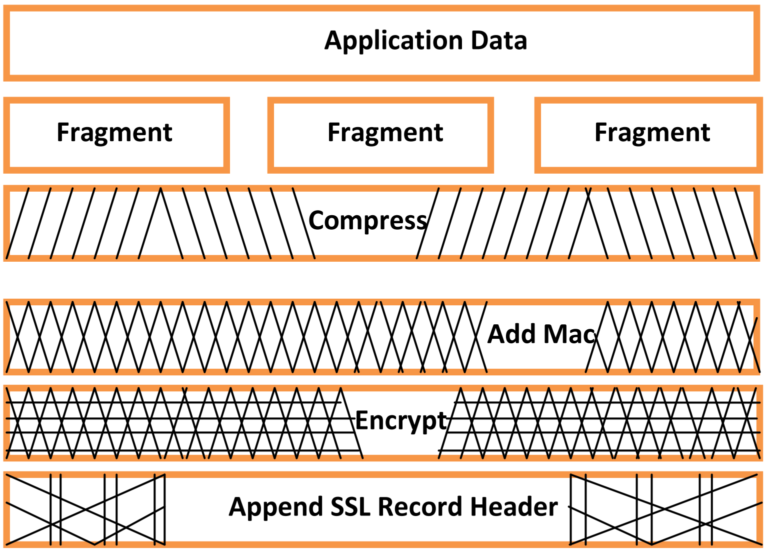


Fig. 6. SSL record protocol

2.11 SSL Limitations

The SSL protocol, like any other technology has its limitations. And because SSL provides security services, it is it is especially important to understand its limits. After all, a false sense of security may be worse than no security. There are 3 categories of SSL limitations. The First are SSL elementary bounds. The tools those are used by SSL protocol, from these tools SSl protocol obtains some imperfections, such as encryption and signature algorithms. The second limitation is that SSL generally cannot rehabilitate them If these algorithms have defects,. Atalast the limit is the environments in which SSL is deployed have their own deficiencies and limitations, in some case like this, here SSL is helpless to address.

2.11.1 Fundamental Protocol Limitations

For many different applications considered by its design, Web transactions with security focused by SSL definitely. For example, TCP as a reliable transport protocol indeeded by SSL. That is a completely reasonable requirement in the world of web transactions, because the HTTP itself requires TCP. The decision means, that SSL cannot operate using a connectionless transport protocol like UDP (User Data Gram Protocol).Another role that SSL fails to fill is support for a particular security service known as non- repudiation. No repudiation associates the digital equivalent of a signature with data.

2.11.2 Tool Limitations

The secure socket layer is simply a communication protocol, and any SSL implementation will rely on other components for many functions, including the cryptographic algorithms. These algorithms are the mathematical tools that actually perform tasks such as encryption and decryption, no SSL implementation can be any stronger than the cryptographic tools on which it is based.

Some common cryptographic algorithms, however, have been successfully attacked. So in general, SSL implementations must consider not only the security of SSL, but also that of the cryptographic services on which it is built.

2.11.3 Environmental Limitations

A network protocol alone can only provide security for information as it transits a network. No network protocol protects data before it is sent or after it arrives at its destination. Security in any computer network, whether the public network or private facilities, is a function of all the elements that make up that network. It depends on the network security protocol, the computer systems that use and the human beings who use those. These SSL protocol is a strong and effective security tool, but it is only a single tool. True security requires many such tools, and a comprehensive plan to employ them.

3. JAVA SECURE SOCKET EXTENSION

100% Pure Java implementation of the SSL and TLS protocols with a framework is provided by The Java Secure Socket Extension (JSSE). The mechanism such as, message integrity, server authentication, ,data encryption, and client authentication as an optional are brought by JSSE. The infatuate about JSSE is that to abstracts the critical highlighting cryptographic algorithms and in this way minimizes the dangerous security vulnerabilities as well as risk of creating subtle. Additionally, To allow users seamlessly integrate SSL into their applications, JSSE develop the secure applications that is quite simple. The JSSE framework is capable of supporting many different secure communication protocols such as SSL 2.0, 3.0 and TLS 1.0.The Java platform's security and encryption features have grown tremendously over the last few years. The JDK 1.4 (a.k.a. Merlin) release now comes bundled with many security related packages, including the Java Cryptography Extension (JCE), the Java Secure Socket Extension (JSSE), and the Java Authentication and Authorization Service (JAAS). These all of the components are fragments of the Java Cryptography Architecture (JCA), as demonstrated in the figure below.

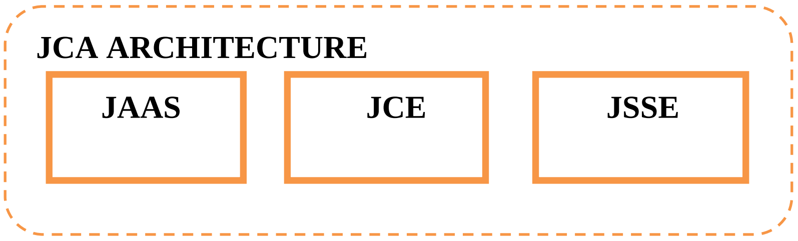


Fig. 7. JCA Architecture

3.1 Programming with JSSE

Java security and java.net packages are supplemented by the JSSE APIs, To epitomize socket generation behavior while affording enhanced networking like a socket classes, key managers, trust and a socket factory framework. In the packages of java such as javax.net and javax.net. ssl, these above classes are included with them.

3.2 JSSE Debugging Utility

Sun's JSSE implementation provides dynamic debug tracing support that can be accessed using the system property javax.net. Debug. This utility, which is not an officially supported feature of JSSE, allows you to see what goes on behind the scenes during the SSL communications. This utility can be used from the command line as follows: Prompt>java -Djavax.net.debug=option [debug Specifies] My SSL App If you use the help option, it will display a list of the debug options.

3.3 Key tool

Key tool is a **key** and **certificate** management utility. It permits users to administrate their own public/private key pairs and for self-authentication (where the users authenticate them to other services) used associated certificates or by using digital signatures also allows for data integrity and authentication services. It also approves users to cache their communicating peers and the public keys (certificates manner). In key store, Key tool stores the keys and certificates. Implementation of default key store resolves the key store as a file. By using password, it protects a private key. The jarsigner tool uses information from a key store to generate or verify digital signatures for Java Archive (JAR) files. (A JAR file packages class files, images, sounds, and/or other digital data in a single file). The digital signature of a JAR file is verified by Jarsigner, by using the certificate that comes with it (in the signature block file of the JAR file, it is included), after this process it checks out the public key of that certificate is "trusted" or not, i.e., is covered in the key store which is specified.

The java key tool provided in JDK 1.1 is completely replaced by the key tool and jarsigner tools. These new tools contains more features on compare to java key, which also include the function to prevent the key store and private keys with the help of strong passwords, it also has the skills to confirm signatures in addition to developing them. The identity of database which is created and managed by java key is replaced by the new key store architecture. To import the information from an identity database into a key store, it is possible but with the help of identity db key tool command. If you use the key tool option in command prompt it will display the key store commands.

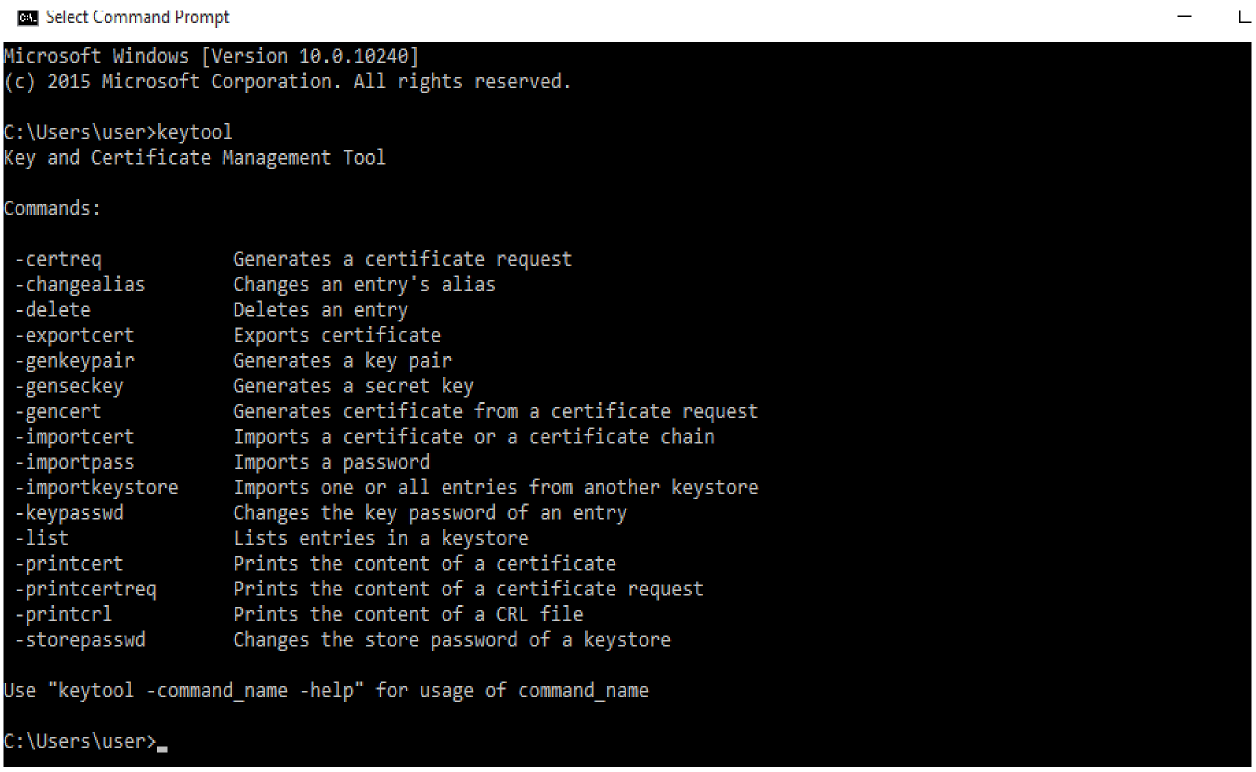

```
Microsoft Windows [Version 10.0.10240]
(c) 2015 Microsoft Corporation. All rights reserved.

C:\Users\user>keytool
Key and Certificate Management Tool

Commands:

 -certreq            Generates a certificate request
 -changealias        Changes an entry's alias
 -delete             Deletes an entry
 -exportcert         Exports certificate
 -genkeypair         Generates a key pair
 -genseckey          Generates a secret key
 -gencert            Generates certificate from a certificate request
 -importcert         Imports a certificate or a certificate chain
 -importpass         Imports a password
 -importkeystore     Imports one or all entries from another keystore
 -keypasswd          Changes the key password of an entry
 -list               Lists entries in a keystore
 -printcert          Prints the content of a certificate
 -printcertreq       Prints the content of a certificate request
 -printcrl           Prints the content of a CRL file
 -storepasswd        Changes the store password of a keystore

Use "keytool -command_name -help" for usage of command_name

C:\Users\user>_
```

Fig. 8 Keytool option list

3.4 Key store Entries
There are two categories of entries in a key store:

Key entries:- Most sensitive cryptographic key information held by the **Key entries**, To prevent from unauthorized access it is stored in a protected format. We can say it is a secret key, which is stored in this type of entity, as well as it can also be known as a private key by the certificate "chain" it is accompanied. On the other way the key jarsigner tools only handle the private keys and their associated certificate chains.

Trusted certificate entries:- Belonging to another party it carries a single public key certificate, that is known as "trusted certificate" because the owner of key store believes that, in the certificate the public key indeed belongs to the identity which is justified by the "subject" (owner) of the certificate. By signing the certificate the issuer of the certificate vouches for this.

3.5 Key store Aliases
With the help of unique aliases all key store entries such as; key and trusted certificate entries, are accessed. Aliases are case-insensitive; The same key store entry referred by the aliases Hugo and hugo. By using the -genkey command to generate a key pair (public and private key) or certificate chain to the list of trusted certificates or the -import command to add a certificate, when you add an entity to the key store, it specifies an alias. Subsequent key tool commands must use this same alias to refer to the entity. For example:- you use the alias duke via the following command: (key tool -genkey -alias duke –key pass dukekeypasswd), to generate a new public/private key pair and wrap the public key into a self-signed certificate(see Certificate Chains).

The alias duke required by subsequent commands to access the private key which is associated with specifies an initial password of "dukekeypasswd. You use a command like the following if you want to change duke's private key password:- (key tool -keypasswd -alias duke -keypass dukekeypasswd -new newpass).

The password will be changed from "dukekeypasswd" to "newpass". Please note: Unless the password is for testing purposes if a password should not actually be specified on a command line or in a script, or you are on a secure system. You will be prompted for it if on a command line you don't specify a required password option. The password is currently echoed (displayed exactly as typed) when typing in a password at the password prompt, so don’t be careless to type it in front of anyone.

3.6 Key store Location
For the key store managed by key tool to specify the name and location of the persistent key store file, each key tool command has a –key store option. As determined by the "user.home" system property, the key store is by default stored in a file named, key store in the user's home directory,

3.7 Key store Creation
In a key store that doesn't exist for adding a data; more specifically if you use a (-genkey, -import, or –identity db) command then key store is created., A key store that doesn't yet exist, that key store will be created if you specify, in the –key store option. On the other hand the default key store is a file named If you don't specify a – key store option. If that file does not yet exist, it will be created then Key stores in your home directory.

3.8 Key store Implementation
Java provides the key store class. To access and modify the information in a key store, well-defined interfaces supplied by the security packages. To implement the multiple different concrete is possible for it, where for a particular type of key store each implementation is used. Currently to use key store implementations, a GUI-based tool named ‘Policy Tool’ and two command-line tools (key tool and jarsigner) are used.

Users of JDK can write additional security applications that use it, as the key store is available publicly,.

Sun Microsystems provides the Built-in default Implementation. The key store as a file implemented by this, Proprietary key store type (format) named "JKS" utilized by it. Each private key as well as an integrity of the entire key store is protected by it with its individual password.

Implementation of key store is provider-based. More specifically, In terms of a "Service Provider Interface" (SPI), Key store supplies the application interface are implemented. There is a KeystoreSpi class is a corresponding abstract, which is also found in the java.security package, It defines the Service Provider Interface methods that "providers" must implement. (The term

"Provider" refers to a package as well as a set of packages through which a concrete implementation of a subset of services supplied that Java Security API access it. In this way, clients must implement a "provider" to provide a key store implementation, and implements a supply of key storeSpi subclass, as described from different provider how to implement a Provider Applications which can select different types of implementations in key store, "GetInstance" factory method is used and supplied in the key store class. The key store information storage and data format is defined by the key store, and with its integrity of the key store itself, the algorithms are used to prevent private keys. Key store implementations of key store in different types are not compatible.

In any file-based key store implementation key store works. (and To converts it to a FileInputStream, from which it loads the key store information it treats the key tore location that is passed to it at the command line as a filename.) On the other hand, the key store from any location that can be specified using a URL is also read by jarsigner and policytool tools. via the - storetype option you can specify a key store type at the command line for key tool and jarsigner. In the edit menu you can specify a key store type via the "Change Key store" command for policy tool,.

The tools choose a key store implementation which is simply based on the value of the key store type whose property specified in the security properties file if you don't explicitly specify a key store type. Java.security is called the security properties file, and where java.home is the runtime environment's directory (the jre directory in the SDK or the top-level directory of the Java2 Runtime Environment), it resides in the JDK security properties directory, java.home\lib\security. Each tools examines all the currently-installed providers until it 25 finds one that implements key store of that type before each tool gets the keystore.type value. From that provider It then uses the key store implementation.

A static method named getDefaultType is defined by the Key store class. The value of the keystore.type property is retrieved by the key store class applications and applets. The following line of code creates an instance of the default key store type (as specified in the keystore.type property): KeyStore key Store = KeyStore.getInstance (KeyStore.getDefaultType());

The "jks" (the proprietary type of the key store implementation provided by Sun) is a default key store. This is specified by the following line in the security properties file: (keystore.type=jks)

Other than the default, a key store implementation to have the tools utilization, to specify a different key store type you can change that line. For example, A key store implementation for a key store type called "pkcs12" which implements a key store and its types if user has a provider package like this. The line to keystore.type=pkcs12 is changed by this package. Note:- In key store type designations this type of case doesn’t matter. For example, "JKS" would be considered the same as "jks"

3.9 Supported Algorithms and Key Sizes

To specify any key pair generation and any of the registered cryptographic service providers supplies signature algorithm allowed by the key store to its user. That is, a provider implementation must support the keyalg and sigalg options for various commands. "DSA" is called as the default key pair generation algorithm. From the algorithm of the underlying private key derives the signature algorithm. The default signature algorithm is "SHA1withDSA" if the underlying private key is of type "DSA", on the other side the default signature algorithm is "MD5withRSA" if the underlying private key is of type "RSA., the key size must be in the range from 512 to 1024 bits, and also a multiple of 64 when generating a DSA key pair. The default key size for any algorithm is 1024 bits. The table that follows explains each option in the command.

Table 1. Debugging options provided by JSSE

OPTION	WHAT IT MEANS
-genkey	Tells key tool to generate a key pair
-alias client private	Identifies the new key pair within the key store
-key store client private	Uses the file client.private as the key store
-storetype JKS	Declares the type of the key store. JKS is the default
-keyalg rsa	Declares the algorithm to be used. Using a RSA algorithm is default.
-dname “CN=Your Name…”	Provide information about the entity owning the key pair
-storepass clientpw	Specifies the password for the entire key store.
-keypass clientpw	Specifies the password for the new key pair

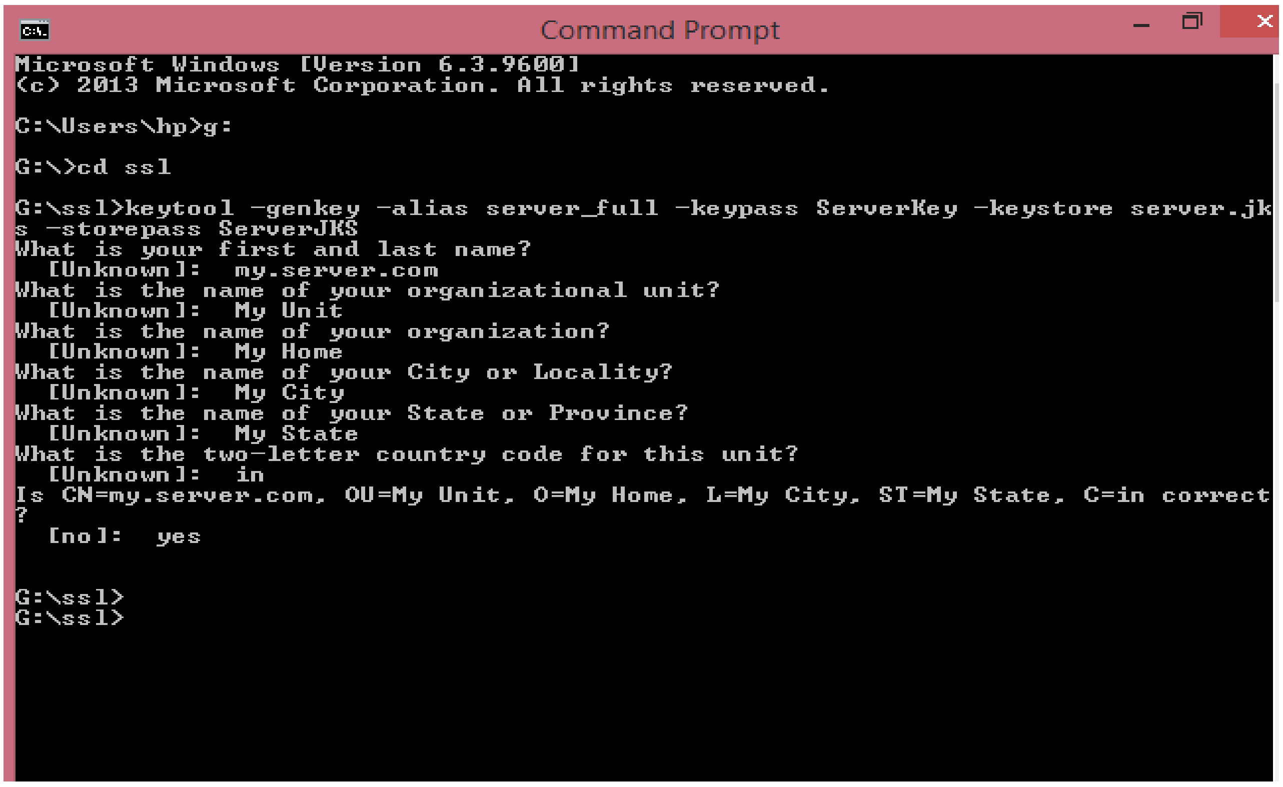
4. SECURE CLIENT SERVER APPLICATION

4.1 Generating a Server Key

To generate and manage certificates JDK “keytool” are the best tool. For both of the sides server and client, “Keytool” is used to generate and manage “full” and “public” certificates. Let's see what I have done on the server side first: >keytool -genkey -alias server full -keypass Server Key -keystore server.jks –storepass Server JKS.

What is your first as well as last name?
[Unknown]: my.server.com
What is your organizational unit name?
[Unknown]: My Unit
What is your organization name?
[Unknown]: My Home
What is the name of your City or Locality?
[Unknown]: My City
What is the name of your State or Province?
[Unknown]: My State
What is the two-letter country code for this unit?
[Unknown]: in
Is CN=my.server.com, OU=My Unit, O=My Home, L=My City, ST=My State...
[no]: yes
>keytool -export -alias server_full -file server_pub.crt -keystore server.jks -storepass
ServerJKS
Certificate stored in file <server_pub.crt>
>"send server_pub.crt to the client side..."

Server's "full" certificate is ready and stored in server.jks at this moment. in server_pub.crt the server's "public" certificate is also ready and stored . For further, let's see what I did on the client side:



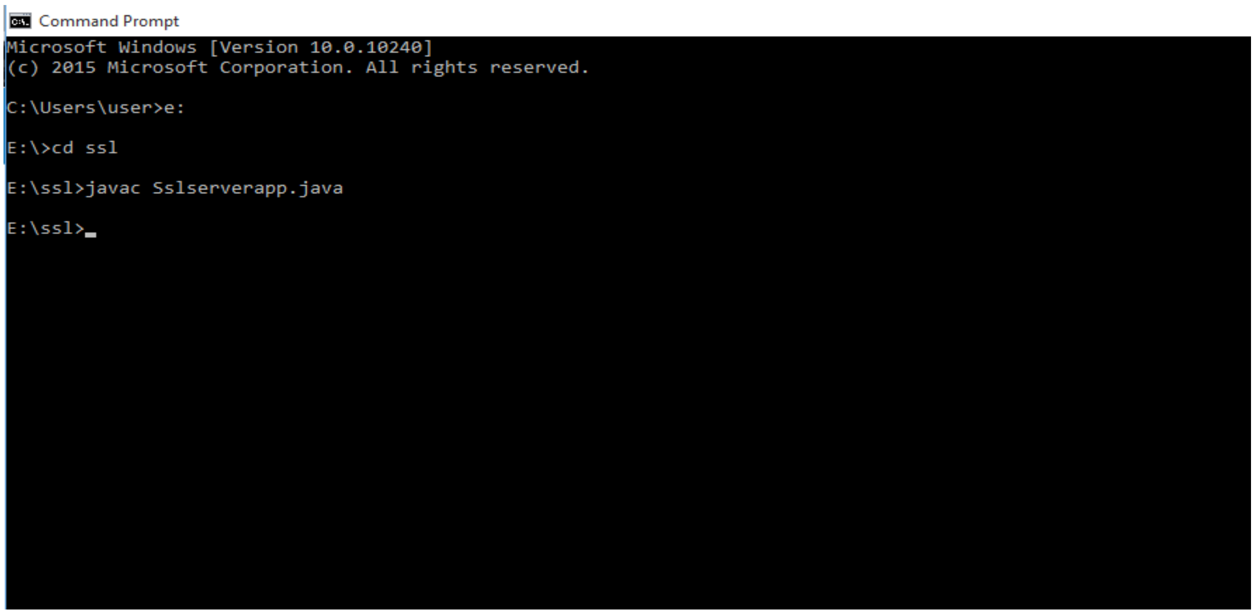
```
Microsoft Windows [Version 6.3.9600]
(c) 2013 Microsoft Corporation. All rights reserved.

C:\Users\hp>g:

G:\>cd ssl

G:\ssl>keytool -genkey -alias server_full -keypass ServerKey -keystore server.jk
s -storepass ServerJKS
What is your first and last name?
  [Unknown]:  my.server.com
What is the name of your organizational unit?
  [Unknown]:  My Unit
What is the name of your organization?
  [Unknown]:  My Home
What is the name of your City or Locality?
  [Unknown]:  My City
What is the name of your State or Province?
  [Unknown]:  My State
What is the two-letter country code for this unit?
  [Unknown]:  in
Is CN=my.server.com, OU=My Unit, O=My Home, L=My City, ST=My State, C=in correct
?
  [no]:  yes


G:\ssl>
G:\ssl>
```

Fig. 9 screen shot for generating a Server key

4.2 Generating a Client key

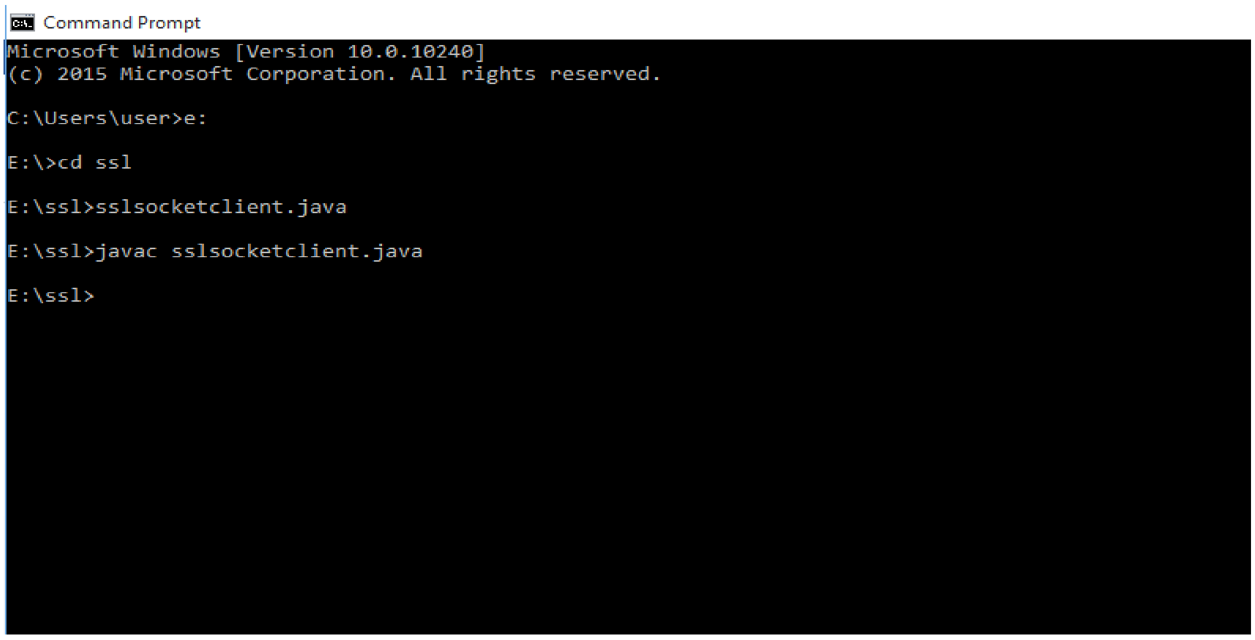
>keytool -genkey -alias client full -keypass ClientKey -keystore client.jks -storepass
ClientJKS
What is your first as well as last name?
[Unknown]: my.client.com
What is your organizational unit name?
[Unknown]: My Unit
What is your organization name?
[Unknown]: My Home
What is your Locality or City name?
[Unknown]: My City
What is your State or Province name?
[Unknown]: My State
For this unit what is the two-letter country code?
[Unknown]: US
Is CN=my.client.com, OU=My Unit, O=My Home, L=My City, ST=My State...
[no]: yes
>keytool -export -alias client_full -file client_pub.crt -keystore client.jks -storepass
ClientJKS
Certificate stored in file <client_pub.crt>
>"send client_pub.crt to the server side..."
>"receive server_pub.crt from the server side..."
>keytool -import -alias cerver_pub -file server_pub.crt -keystore client.jks -storepass
ClientJKS
Owner: CN=my.server.com, OU=My Unit, O=My Home, L=My City, ST=My S...

Issuer: CN=my.server.com, OU=My Unit, O=My Home, L=My City, ST=My ...
......
Trust this certificate? [no]: yes
Certificate was added to keystore
As you can see, the keytool prompted me to enter a password for the keystore meaning that in order for the server to access the keystore it must know that password. Also, the tool asked me to enter a password for the alias. If you like, such password information can be related on the keytool command line using the options -storepass and -keypass. Note that I used the name "ultra.domain.com" for the first and last name. This name is the hypothetical name of my machine. You should enter the hostname or the IP address of the server's machine. When you run the keytool command, it may take a few seconds to generate the certificate depending on the speed of your machine. Once a certificate for my server has been generated, I can revise my Http Server to make it secure. If you examine the Http Server class, you'll notice that the get Server method is used to return a server socket. This means, the only method I need to modify is the getServer method so that it returns a secure server socket. The changes are highlighted in bold in Code Sample 2. Notice that I have changed the port number to 443. This is the default port number for HTTPs. It is important to note that port numbers between 0 and 1023 are reserved. If you run Https Server on a different port number, the url should be: https://localhost:portnumber but if you run it on443 then the URL is: (https://localhost)



```
Microsoft Windows [Version 10.0.10240]
(c) 2015 Microsoft Corporation. All rights reserved.

C:\Users\user>e:

E:\>cd ssl

E:\ssl>javac Sslserverapp.java

E:\ssl>_
```

Fig. 10 Compilation of Server code

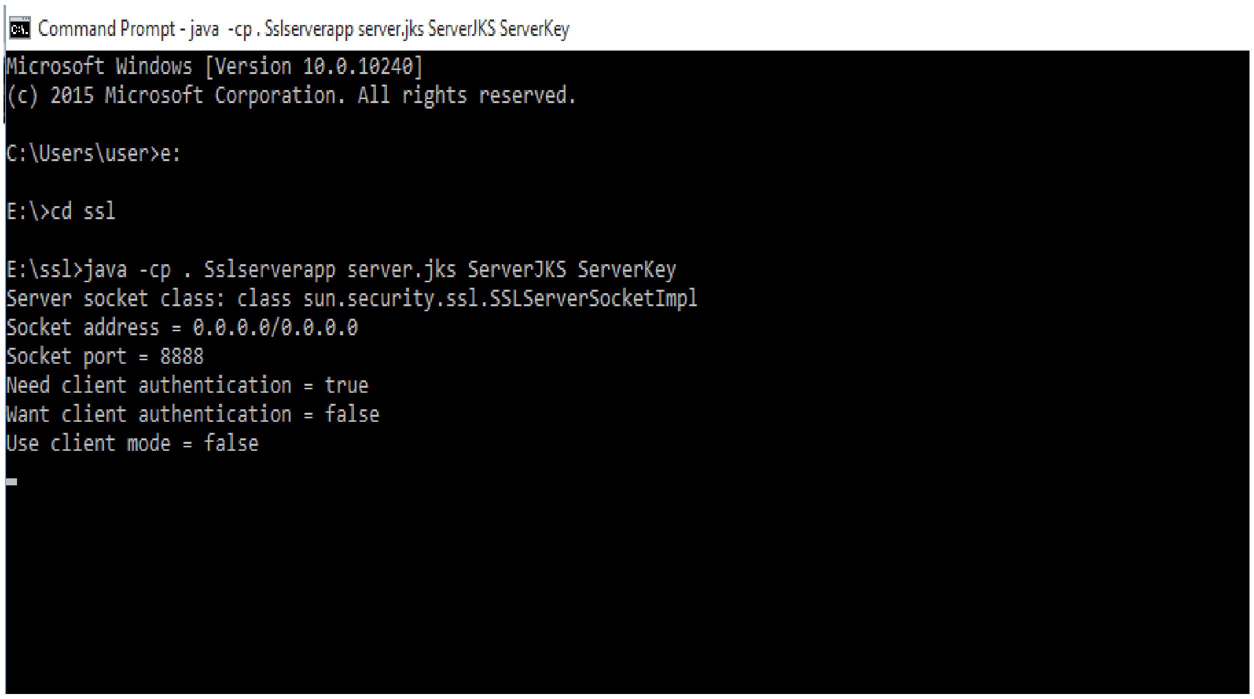


```
Microsoft Windows [Version 10.0.10240]
(c) 2015 Microsoft Corporation. All rights reserved.

C:\Users\user>e:

E:\>cd ssl

E:\ssl>sslsocketclient.java

E:\ssl>javac sslsocketclient.java

E:\ssl>
```

Fig. 11 Compilation of client application

4.3 SSL Client Authentication Test Result

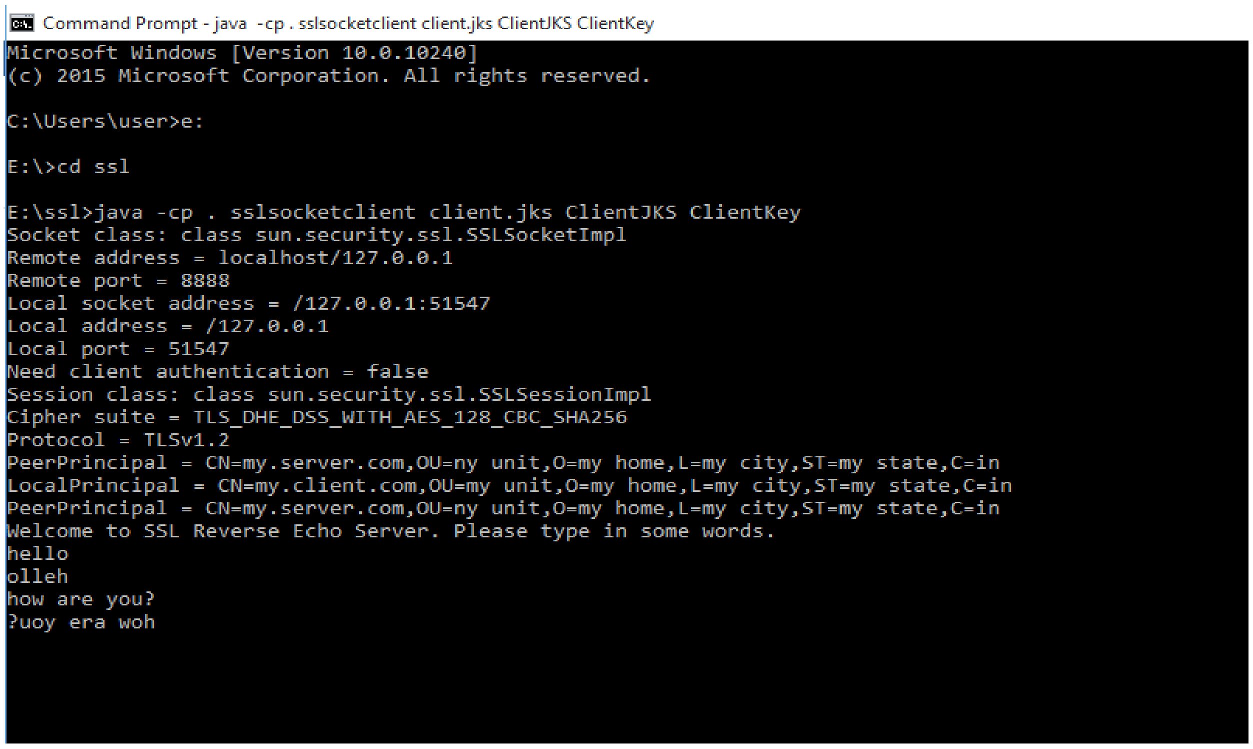
With Sslserverapp.java and server.jks prepared on the server side, I am ready to start the server program:



```
Command Prompt - java  -cp . Sslserverapp server.jks ServerJKS ServerKey
Microsoft Windows [Version 10.0.10240]
(c) 2015 Microsoft Corporation. All rights reserved.

C:\Users\user>e:

E:\>cd ssl

E:\ssl>java -cp . Sslserverapp server.jks ServerJKS ServerKey
Server socket class: class sun.security.ssl.SSLServerSocketImpl
Socket address = 0.0.0.0/0.0.0.0
Socket port = 8888
Need client authentication = true
Want client authentication = false
Use client mode = false
```

Fig.12 Compilation of server code

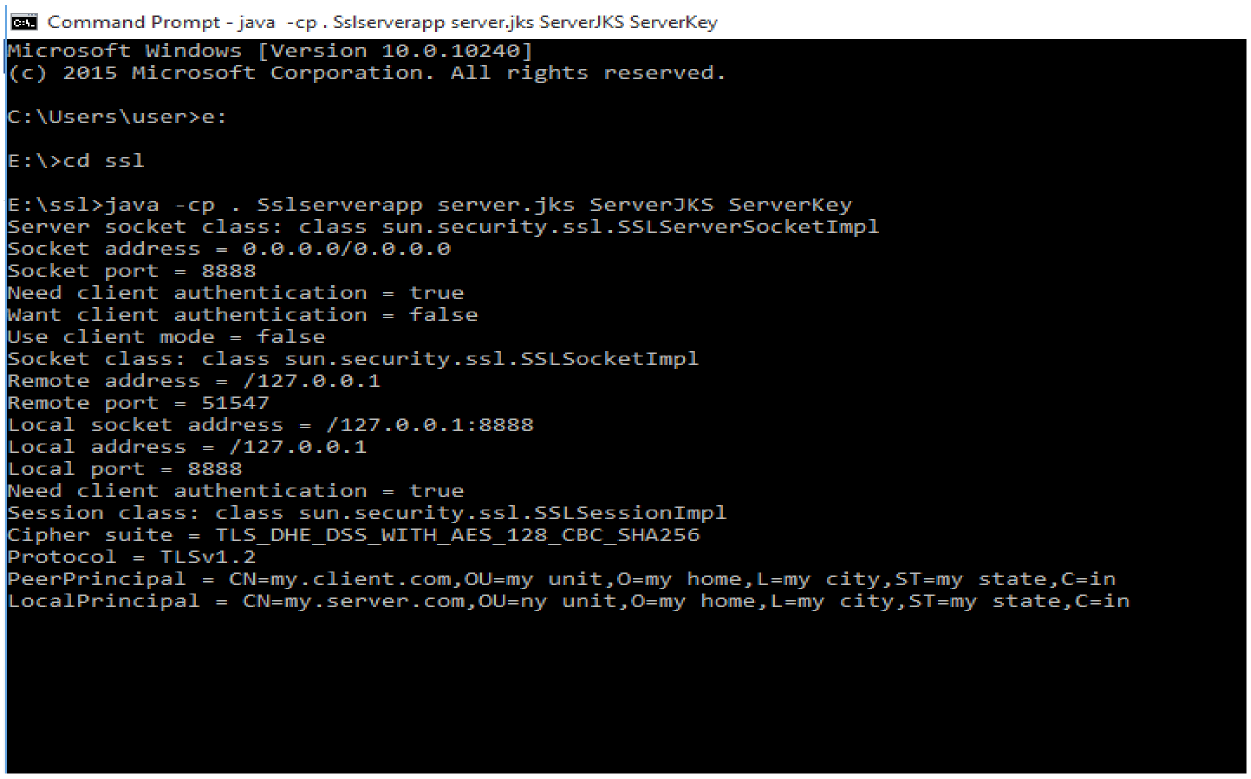
Now switch to the client side, and run the client program



```
Command Prompt - java  -cp . sslsocketclient client.jks ClientJKS ClientKey
Microsoft Windows [Version 10.0.10240]
(c) 2015 Microsoft Corporation. All rights reserved.

C:\Users\user>e:

E:\>cd ssl

E:\ssl>java -cp . sslsocketclient client.jks ClientJKS ClientKey
Socket class: class sun.security.ssl.SSLSocketImpl
Remote address = localhost/127.0.0.1
Remote port = 8888
Local socket address = /127.0.0.1:51547
Local address = /127.0.0.1
Local port = 51547
Need client authentication = false
Session class: class sun.security.ssl.SSLSessionImpl
Cipher suite = TLS_DHE_DSS_WITH_AES_128_CBC_SHA256
Protocol = TLSv1.2
PeerPrincipal = CN=my.server.com,OU=ny unit,O=my home,L=my city,ST=my state,C=in
LocalPrincipal = CN=my.client.com,OU=my unit,O=my home,L=my city,ST=my state,C=in
PeerPrincipal = CN=my.server.com,OU=ny unit,O=my home,L=my city,ST=my state,C=in
Welcome to SSL Reverse Echo Server. Please type in some words.
hello
olleh
how are you?
?uoy era woh
```

Fig. 13 Compilation of client code

Looking at the server side again, you will see messages:-



Command Prompt - java -cp . Sslserverapp server.jks ServerJKS ServerKey

```
Microsoft Windows [Version 10.0.10240]
(c) 2015 Microsoft Corporation. All rights reserved.

C:\Users\user>e:

E:\>cd ssl

E:\ssl>java -cp . Sslserverapp server.jks ServerJKS ServerKey
Server socket class: class sun.security.ssl.SSLServerSocketImpl
Socket address = 0.0.0.0/0.0.0.0
Socket port = 8888
Need client authentication = true
Want client authentication = false
Use client mode = false
Socket class: class sun.security.ssl.SSLSocketImpl
Remote address = /127.0.0.1
Remote port = 51547
Local socket address = /127.0.0.1:8888
Local address = /127.0.0.1
Local port = 8888
Need client authentication = true
Session class: class sun.security.ssl.SSLSessionImpl
Cipher suite = TLS_DHE_DSS_WITH_AES_128_CBC_SHA256
Protocol = TLSv1.2
PeerPrincipal = CN=my.client.com,OU=my unit,O=my home,L=my city,ST=my state,C=in
LocalPrincipal = CN=my.server.com,OU=ny unit,O=my home,L=my city,ST=my state,C=in
```

Fig. 14 Result of server code

Client program authenticated the server's identity ok, and server program authenticated the client's identity ok too.

5. CONCLUSION

This paper was implemented using the Secured Sockets Layer Protocol in JAVA. In this project, we created an encrypted link and demonstrated a message transfer through this link between the server and a client using Raviest Shamir Adleman Encryption standards. SSL protocol provides total security to data which is sent over the communication network. The analysis shows that it provides different measures of security against eavesdropping and other passive attacks. The analysis has also revealed several ways in which the robustness of the SSL protocol can be improved. The usage of SSL in securing the data transmitted through web pages. SSL is most common security protocols SSL protocol used in the Internet for facilitating secure communications through authentication, encryption, and decryption. Application is developed to overcome the problem of security and authentication. The conclusions are presented below:

- There are two side of this application: server side and client side. Without certification, client can't communicate with the serve. The SSL handshake-protocol message flow involves client and server negotiating a common cipher suite acceptable to both parties.
- The application based on RSA algorithm, using for encryption especially for data sent to server.
- Keystore file, extension with .jks (java key store) holds very sensitive cryptographic key information, which is stored in a protected format to prevent unauthorized access.
- This application is for secure gateway for string messages.

6. REFERENCES

[1] A. O. Freier P. Karlton and P. C. Kocher. The SSL Protocol, Version 3.0. Netscape Communications, 1996, http://wp.netscape.com/eng/ssl3/draft302.txt (2003).

[2] R. Rivest. The MD-5 Message-Digest Algorithm, RFC 1321. John Wiley and Sons, 1996, ftp://ftp.rfc-editor.org/innotes/rfc1321.txt (2003).

[3] Charlie Scott, Paul Wolfe, and Mike Erwin, "Virtual Private Networks",Publisher: O'Reilly, Second Edition, ISBN: 1- 56592-529-7, pp.6-40.

[4] R. Rivest, A. Shamir, and L. M. Adleman, "A Method for Obtaining Digital Signatures and Public-Key Cryptosystems ", Communications of the ACM, v. 21, n. 2, Feb 1978, pp. 120-126.

[5] Stephen Thomas, "SSL and TLS Essentials, Securing the Web",Publisher: Wiley, pp.12-14 & 37-60.

[6] Giacomazzi, Poli, A. " Cost-Performance Optimization of SSLBased Secure Distributed Infrastructures"in Latin America Transactions, IEEE (Revista IEEE America Latina) Volume: 9 , Issue: 4 , Page(s): 550 – 556 in 2011.

[7] C. J. Lamprecht, Aad and P. A. van Moorsel "Adaptive SSL: Design, Implementation and Overhead Analysis " in IEEE Computer Society, July 2007.

[8] Norman Lim, Shikharesh Majumdar, Vineet Srivastava "Engineering SSL-based systems for enhancing system performance" in Proceedings of the 2nd ACM/SPEC International Conference on Performance engineering, March 2011.

[9] Kapil Singh, Helen J. Wang, Alexander Moshchuk, Collin Jackson, Wenke Lee "Practical end-to-end web content integrity" in Proceedings of the 21st international conference on World Wide Web in ACM, April 2012.

[10] Martin Georgiev, Subodh Iyengar, Suman Jana, Rishita Anubhai, Dan Boneh, Vitaly Shmatikov "The most dangerous code in the world: validating SSL certificates in non-browser software " Proceedings of the 2012 ACM conference on Computer and communications security ,October 2012.

[11] Light, J. Ikejiani, O.K. "An efficient wireless communication protocol for secured transmission of content-sensitive multimedia data " in World of Wireless, Mobile and Multimedia Networks & Workshops IEEE Page(s): 1 – 6, April 2009

[12] Norazah Abd Aziz, Nur Izura Udzir and Ramlan Mahmod,Performance Analysis for Extended TLS with Mutual Attestation for Platform Integrity Assurance,IEEE,2014.

[13] LI Wei, XIANG Shuyue, CHEN Shuangbao, Improvement Method of SSL Protocol Identity Authentication based on the Attribute Certificate, International Conference on Computer Science and Service System,2012.