Web Payment Security

A discussion of methods providing secure communication on the Internet

Group Members:

Peter Heighton

Zhao Huang

Shahid Kahn

1. Introduction

Within this report the methods taken to ensure communications on the Internet are secure will be discussed, with particular reference to e-commerce transactions. Today, we think of e-commerce as business conducted electronically, over a network such as the Internet. With the consumers growing need for 'convenience', e-commerce is now a very large business, with services including online banking and shopping, there are very few things that cannot be obtained via the Internet. Web commerce more precisely refers to the payment and purchase of goods over the World Wide Web, but there are still concerns with the security of such transactions. With more and more reports of identity theft, where customer details are obtained and can then be used to make purchases, what measures are taken to ensure such valuable details can be transferred securely and how can a customer be sure of security?

2. HTTPS

Lets first look at the underlying protocols in place for secure transactions. *HTTPS* is an application layer protocol identical to *HTTP*, but using port 443 (instead of port 80) and an additional encryption layer (SSL/TLS), for security. The user knows that data is transmitted securely when a closed lock appears in the web browser, usually located in the lower right or left corner. The user can double-click the lock icon to view the sites security information. An additional check can be made by checking the URL, which should begin with *https.*

2.1 SSL/TLS

Secure Sockets Layer (SSL) is used in conjunction with *http,* to provide the *https* protocol. SSL is a cryptographic protocol used to ensure the secure transfer of data. This means that any data sent using SSL is encrypted as opposed to being sent in 'plain text', that anyone can read should it be intercepted. Since there is no way of telling which route data may take, the possibility of interception is very real. An additional problem is that a site may be pretending to be a well-known corporation, but in reality has nothing to do with them, so the user needs *authentication* to prove it is the intended destination. SSL works by encrypting the data before it is sent across the World Wide Web, and also provides authentication between client and server. An initial handshake, much like TCP connection establishment, occurs to agree the encryption algorithm (e.g. DES, IDEA…) and transfer keys. This process occurs to authenticate the server to the client. Once an authentic connection has been established, the public key is used for encryption, and the destination server, to decipher the data, uses the private key. SSL actually lies between the *application* layer and the *transport* layer. How this works is that, 'on the sending side, SSL receives an *http*

message from an application, encrypts the data, and directs the encrypted data to a TCP socket. On the receiving side, SSL reads from the TCP socket, decrypts the data, and the data on the application.' Ref: Pg 709 Kurose & Ross.

2.2.1 Summary of Features:

- *SSL server authentication*
  User is assured of server's identity by use of certificates. Trusted certification authorities (CAs) are listed in the users web browser containing their public keys. When an SSL connection is made the browser obtains a certificate from the server. If the certificate matches a trusted authority, communication is granted.

- *SSL client authentication*
  Mutual authentication can also be granted if necessary, allowing the server to identify the user. The server will, in this case, obtain certificates from the client, for authentication.

- *Encrypted SSL session*
  All data sent between client and server is encrypted and decrypted upon receipt. Tampering of the data can also be detected by SSL.

2.2 How SSL Works

So what actually happens when a user submits information from an *https* web page? As previously mentioned, the SSL handshake protocol is initiated which authenticates the server, using public key techniques. The transfer of encrypted data makes use of shared symmetric keys. The combination of techniques is used since public key encryption is better for authentication, but symmetric key encryption is much faster and suited to the rapid encryption/decryption necessary during the communication session. Specific detail of the handshake are as follows:

1. Client sends server its SSL version number and cryptographic information necessary to decide on the symmetric key algorithm to be used.
2. Server sends client its SSL version number, necessary cryptographic information, along with its certificate containing the server's public key, and signed with the CA's private key.
3. The client checks its list of trusted CA's to find a match. If no match is apparent then the client is informed that an authenticated connection cannot be established. If there is a match, the certificate is validated with the client's public key for the CA. The client can now obtain the server's public key.

4. The client then generates a ‘pre-master secret’, encrypted with the server’s public key. If the server has requested client authentication, the client will send the symmetric key along with its own certificate.
5. If the server cannot authenticate the client, the session ends. If authentication is successful, the server’s private key is used to decrypt the ‘pre-master secret’. Both server and client using the ‘pre-master secret’ then generate a ‘master secret’, from which the symmetric session keys are generated.
6. The client informs the server that all future communication will be encrypted with the session key. A separate, encrypted message is sent notifying its handshake has ended.
7. The server informs the client that all future communication will be encrypted with the session key. A separate, encrypted message is sent notifying its handshake has ended.
8. The handshake is now complete, and the session can begin, using the session keys for encryption, decryption and validity.

2.3 Where does TLS fit in?

Transport Layer Security (TLS) is the successor to SSL, based on Netscape’s SSL 3.0 (most current version). TLS 1.1 is the most current version of the protocol and is supported by most recent web browsers, such as Opera and GnuTLS software. Though very similar, TLS and SSL 3.0 cannot be used in conjunction with one another.

2.4 Why are there still Vulnerabilities?

OK so we are no longer susceptible to man-in-the-middle attacks, and it would appear that sensitive data is completely safe traveling across a network. So why do fraudulent attacks still occur? The answer can lie in the SSL version. Although SSL 3.0 has been around since 1996, web browsers such as Internet Explorer and Firefox still continue to provide compatibility with SSL 2.0 as default. Now vigilant users that look out for the ‘golden lock’ symbol may find their information is being sent over a ‘secure’ connection, but using SSL 2.0 with much weaker cryptography. In reality very few websites offer SSL 2.0, but to be safe it is worth disabling the connection.
Another mistake often implemented by developers is to not have an initial *https* login page, to increase efficiency and convenience. The idea is that only sensitive data is sent via *https.* The first simple problem is that users do not know whether the form is being submitted via *https* and if they do is it going to the expected page? This design opens up the man-in-the-middle attack since the login form was sent via *http* it could be intercepted and sent to a ‘bad’ *https*

page. Another more complex problem is that an attacker could re-write the login page and record log the users keystrokes. A final mistake made by developers is using mixed *http* and *https* content on the same page. This makes it impossible to tell which parts have been delivered securely and again invites interception, with consequences such as an attacker re-writing the *https* page or obtaining sensitive information.

3. Cookies

When I visit certain websites for shopping online, they always show “Hi, zhao”, or a similar greeting. This makes me very confused and worried. How do these websites know me?

Some websites know you because at some time in the past you registered with the site—most likely when you bought something. The site then deposited a small text file called a cookie on your computer's hard disk. Cookies work like a kind of ID tag. Every time you visit the site, it looks for the cookie, reads the text string inside, and then uses it to call up your record in the website database which you are browsing. The site then loads pages containing your preferences, billing address (if you've provided that), new products that you might be interested in buying (based on past purchases), and that friendly greeting at the top of the screen. Certainly, if you set your browser not to accept cookies, then the websites won’t know who you are. Some online shops prefer doing it like this to *appear* more intimate with you.

4. Payment Types

When I make purchases on the net, I don’t feel as secure as the traditional way of shopping. I am worried that someone is spying on my account, my credit card number, password or some other sensitive and private information. Is there any suggestion to heal these worries?

Frankly speaking, your card number can be stolen even when shopping at secure sites. For example, someone could install a keystroke logger on your computer. What’s worse, the e-commerce site could be hacked as well. Thus the solution: get a disposable credit card number. Nowadays, some banks use a verification code, which can only be used once, binding the credit card number and password that is good for a single transaction, so even if the number is spied out, it won't do thieves much good. This scheme is similar to the nonce in the digest HTTP authentication protocol to some extent. Another alternative is to get a separate credit card with a low limit and use it exclusively for online purchases. Thus you may minimize your loss.

Now, we are talking about several primary types of payment when we purchase on the net.

4.1 Credit card.

As you can see, nowadays most online transactions are done with Credit Cards. Since this paradigm does not need much for infrastructure and can be understood easily by users.

Although using a credit card to purchase on the net is very popular, there are still some flaws of credit card payment:

For example:

- Credit card Frauds
- No customer signature
- Legitimacy of merchant
- Privacy is always compromised with spam emails and telemarketing calls
- Sometimes, the cost of transferring a payment can exceed the cost of the product itself.

4.2 Electronic Cash

Electronic Cash act as a form of data. It transforms the cash value to an encrypted serial number and represents the cash we use normally. In other words it holds cash attributes, such as it can be saved, withdrawn and transferred.

4.2.1 How does the Electronic Cash work?

Accurately speaking, an Electronic cash coin begins life as a 100-digit random number chosen by a user software package .To use the Electronic Cash, first the user needs to deposit the amount of E-cash by software and save it in his hard disk. Then he transmits this number to the bank along with the denomination requested. If the bank verifies that the number is not already in use, it validated the number by encoding the number with the private key and debits the user's account. The bank transmits the validated number back to the user, who records the coin on his hard disk. Meanwhile, it tracks the “serial number” of each coin that it issues to ensure that a coin is not used twice. Merchant must immediately check with the bank to verify the coins for each and every transaction

Thus as mentioned above, since the process of using Electronic Cash is not simple enough, this is computationally expensive for buying something like a cup of coffee and so forth.

4.3 SET (Secure Electronic Transaction)

SET, short for Secure Electronic Transaction, is a standard that will enable secure credit card transactions on the Internet. SET has been endorsed by virtually where complement the drawbacks of simple credit card payment scheme.

4.3.1 Summary of several primary techniques used in this mechanism:

- Based on iKP
- Developed by a consortium led by MasterCard and Visa
- Digit Signature: Uses RSA for signatures
- Private-key encryption: DES

4.3.2 The goal of SET protocol: SET

①┌ To ensure(Guarantee) the confidentiality of the information between the participants because the information is only available to parties in a transaction when and where necessary.
②┌ To avoid the information be spy out as it travels across the network. Providing a secure communications channel among all parties involved in a transaction.
③┌ Give a solution to multi-authentication
④┌ Realtime transaction
⑤┌ Regulate protocol and message form in order to be adapted readily to the different operate systems

4.3.3 The participants refer to the SET protocol SET

⑴ Customer

⑵ Online shop. Provide products or services, meanwhile it has the capability of using electronic cash.

⑶ Bank. Deal with the payment problem between customs and online shop by payment gateway.

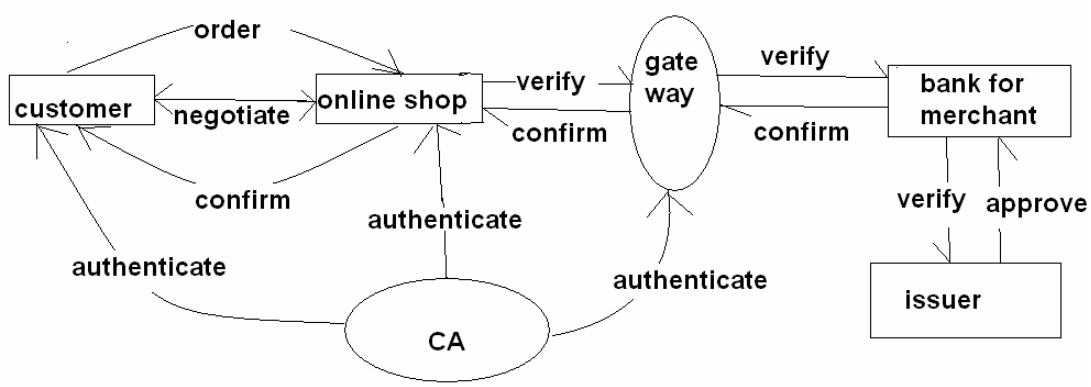
⑷ Electronic money Issuer or some banks which can issue the E-money These financial institution are responsible for the payment of debt of the customs.

⑸Certificate Authority. This is an entity that is trusted to issue X.509v3 public – key certificates for customs, online shops and so forth. In charge of authentication for payment method and identification of participants.

So we can conclude several features of SET as following:

(1) Primarily deals with credit card payments
(2) Information confidentiality
(3) Data integrity
(4) Authentication of customer's account
(5) Verify of online shop
(6) Interactive operation

SET regular the e-commerce payment transaction flow (see the attachment):



The participants respectively are: customers, merchant and payment gateway. The flow can be described as following:

（1） The customer sent the order to merchant with a list of items to be purchased(/online shop)

（2）The merchant return message to custom so that the customer can check if he is dealing with a valid store

（3）Customer verify the merchant authentication and sent the order along with the payment information. However, the payment information including credit card details is encrypted in such a way that can not be read by the merchant (use the share-key of bank to encrypt)

（4）The merchant verify the gateway authentication and sent the payment information to the payment gateway, requesting authorization that the customer's available credit is sufficient for this purchase.

（5）The payment gateway confirm the authentication of merchant and the valid of payment information. Then the gateway responds with the result, to the merchant.

（6）The merchant sent the information to the customer along with the stuffs according to the order which the customer sent to the merchant ealier on the net.

（7）The merchant regularly sent the payment information to the payment gateway . The gateway informs the bank sent the money to the merchant account as soon as it receives the payment information from the merchant. After doing these above, the gateway return the result to the merchant. Thus the transaction is over.

Since the implementation of the SET scheme is rather complex. It needs the encryption, digital signature and the software both fixing on the client and server and so forth for every transaction. So there are not many websites adopting this mechanism while the bank of china (which is the most authority bank in China) used the SET for payment on the net.

References

Computer Security – D.Gollman

Computer Networking 3rd Ed – J.Kurose & K.Ross

http://support.microsoft.com/default.aspx?scid=kb;en-us;257591 (23/10)

http://blog.ziffdavis.com/serveriqsecurity/archive/2006/03/27/40543.aspx (23/10)

http://blogs.msdn.com/ie/archive/2005/04/20/410240.aspx (23/10)

http://news.netcraft.com/archives/2006/05/31/most_sites_ready_for_ssl_progress.html (23/10)

http://info.ssl.com/article.aspx?id=10068 (23/10)

http://searchsecurity.techtarget.com/sDefinition/0,,sid14_gci557332,00.html (23/10)

http://en.wikipedia.org/wiki/Secure_Sockets_Layer (23/10)