

Defeating Active Phishing Attacks for Web-Based Transactions

Xin Luo, Virginia State University, USA

TAN Teik Guan, Data Security Systems Solutions Pte Ltd, Singapore

ABSTRACT

Till now, the best defense against phishing is the use of two-factor authentication systems. Yet this protection is short-lived and comparatively weak. The absence of a fool-proof solution against man-in-the-middle, or active phishing, attacks have resulted in an avalanche of security practitioners painting bleak scenarios where active phishing attacks cripple the growth of Web-based transactional systems. Even with vigilant users and prudent applications, no solutions seem to have addressed the attacks comprehensively. In this article, we propose the new two-factor interlock authentication protocol (TIAP), adapted from the interlock protocol with two-factor authentication, which is able to defend successfully against active phishing attacks. We further scrutinize the TIAP by simulating a series of attacks against the protocol and demonstrate how each attack is defeated.

Keywords: interlock protocol; phishing; two-factor authentication

INTRODUCTION

The current wave of phishing attacks against Internet banking and transaction Web sites is only the tip of the hacking iceberg in the field of information systems security. Yet, these relatively unsophisticated attacks have already catastrophically resulted in significant monetary loss and are a major source of embarrassment to the financial

institutions. This predicament has drawn increasing attention from both security researchers and practitioners. Early research has shed light on such tactical antiphishing methods as having Internet service providers (ISPs) involved in closing phishing Web sites and launching retaliatory services to proactively block phishing traffic. However, these approaches are time-consum-

ing and expensive, and are even useless in countries that lack relevant antiphishing regulations (Geer, 2005). While organizations are scrambling to deploy costly two-factor authentication solutions (i.e., having a one-time password in addition to a normal password) to cope with the problem, such remedies may just be short-lived as the hackers can easily deploy the more sophisticated active phishing attacks to thwart the security and the additional effort could cause consumers to avoid Internet banking (Geer, 2005).

Defined as attacks that use both social engineering and technical subterfuge to steal consumers' personal identity data and financial account credentials (Goth, 2005), phishing incidents have gradually eroded consumer confidence in online banking (Geer, 2005) and further imposed immeasurable losses for corporations in terms of time and resources. In addition to public education, authentication such as one-time password technology may be successful at preventing off-line or static phishing attacks (Bellovin, 2004). While researchers have previously addressed the technological concerns of static phishing and proposed relevant solutions such as phishing Web page detection based on visual similarity (Liu, Huang, Liu, Deng, & Zhang, 2005), mail filtering methods (Inomata, Rahman, Okamoto, & Okamoto, 2005) and XUL and JavaScript-based browser extensions (Kirda & Kruegel, 2005), the field of active phishing is still unexplored. The possibility of active phishing or online man-in-the-middle attacks has been troubling security practitioners and consultants (Schneier, 2005) for a while already. In general, active phishing can be defined as the use of a reverse proxy in the middle to dynamically access the actual site while phishing the user, thus giving the impression that the

user is communicating with the correct site, while the hacker in the middle has actual control of the session and may modify the contents to achieve illegitimate gains.

Along with Herzberg's argument that SSL/TLS is limited and weak for site impersonation and scam sites (Herzberg, 2004), we believe that the difficulty in preventing active phishing attacks for Web-based transactions is due to the fact that the HTTP-over-SSL protocol is easily reverse-proxied. In fact, all SSL-VPN solutions exploit this reverse-proxy capability somewhat to support a seamless VPN tunnel between the browser through the SSL-VPN gateway to the backend application server. Hence, the SSL-VPN gateway is in fact functioning as a "good" man-in-the-middle to provide the VPN encryption functionality.

The problem is further acerbated by the inherent fact that the client executable content (i.e., the HTML/Javascript in the browser) is actually downloaded from the server. This means that the server with which the browser is communicating has full control over whatever content is executed on the browser. Should the browser be communicating with a phishing server, there is no way that the actual server is able to bypass this problem.

A security-conscious bank with a security-conscious user base does not guarantee that the Internet banking sessions between them are secure. Already, a case of active phishing has been reported (Kirk, 2005), and it is only a matter of time before the exploitation of the vulnerability becomes widespread.

THEORETICAL BACKGROUND

The interlock protocol by Rivest and Shamir (1984) is an elegant solution designed to defeat hackers who attempt to eavesdrop

on communication. In Figure 1, Alice and Bob are using the Diffie-Hellman exponential key exchange protocol (Diffie & Hellman, 1976) to establish a shared secret key which can be used subsequently to encrypt the session. The Diffie-Hellman exponential key exchange works by first having a large public value n , used as the modulus, a public random value g and a secret random value a generated by Alice. Alice computes $g^a \text{ mod } n$ and sends over the result, along with g and n (a remains secret in Alice's possession). Bob, would generate a secret random value b , compute $g^b \text{ mod } n$, and return the result of the modulo exponent to Alice. Both Alice and Bob are able to compute $g^{ab} \text{ mod } n^1$, thus arriving

at a shared secret key k , which no one else is able to cryptographically derive. Using the shared secret key k , Alice would be able to encrypt any secret information, denoted as $E_k(\dots)$, and send it securely to Bob, with certainty that only Bob is able to decrypt it, and vice versa. However, the communication is easily susceptible to man-in-the-middle attacks.

Mallory in Figure 2, will perform a key exchange with both Alice and Bob, posing as the opposite party to obtain a shared secret key k with Alice, and a different shared secret key k' with Bob. Whatever information that Alice encrypts using k can be decrypted by Mallory and re-encrypted using k' to be forwarded to Bob, thus achieving the man-in-the-middle attack.

In this scenario, the secret information encrypted by the session key is accessible by Mallory, and subject to illegal modifications. The interlock protocol breaks the man-in-the-middle attack by destroying Mallory's ability to decrypt using one key, and re-encrypt using a different key in real time. After Mallory has successfully negotiated the secret keys, the trick in the interlock protocol is that it requires that the information encrypted by the session key

Figure 1. Exponential key exchange

| Alice (user) | | Bob (server) |
|-----------------------------|---|-----------------------------|
| $g^a \text{ mod } n$ | → | |
| | ← | $g^b \text{ mod } n$ |
| $k = g^{ba} \text{ mod } n$ | | $k = g^{ab} \text{ mod } n$ |
| $E_k(\text{secret info}_a)$ | → | |
| | ← | $E_k(\text{secret info}_b)$ |

Figure 2. Man-in-the-middle attack on exponential key exchange

| Alice | | Mallory | | Bob |
|-----------------------------|---|-------------------------------------------------------------|---|--------------------------------|
| $g^a \text{ mod } n$ | → | | | |
| | ← | $g^m \text{ mod } n$ | | |
| | | $g^m \text{ mod } n$ | → | |
| | | | ← | $g^b \text{ mod } n$ |
| $k = g^{ma} \text{ mod } n$ | | $k = g^{am} \text{ mod } n$ $k' = g^{bm} \text{ mod } n$ | | $k' = g^{mb} \text{ mod } n$ |
| $E_k(\text{secret info}_a)$ | → | $E_{k'}(\text{secret info}_a)$ | → | |
| | ← | $E_k(\text{secret info}_b)$ | ← | $E_{k'}(\text{secret info}_b)$ |

be divided into two halves², and each half is sent after receiving a half from the other party (see Figure 3).

This method defeats Mallory’s attempt at the man-in-the-middle attack on the onset of the communication, since Mallory only has received half of the secret info and thus is unable to determine what information, denoted by $E_k(??)$, to send to the other party. He is at best able to receive 1 proper message from either Alice or Bob, but is unable to continue the conversation between Alice and Bob.

Positive as it may seem, the interlock protocol is not entirely suitable for preventing man-in-the-middle authentication attacks. Bellovin and Merritt (1992) proposed that by doing a slightly more sophisticated attack, Mallory is able to steal Alice’s authentication credentials and open a session with Bob (and even potentially stealing Bob’s authentication credentials).

Additionally, the other acknowledged solution is the encrypted key exchange

(EKE). The EKE by Bellovin and Merritt (1994) was designed initially to address the frequent use of poorly chosen passwords used during authentication. By using these passwords to encrypt a randomly generated public key pair, the EKE achieves the ability to prevent the poorly chosen passwords from being easily guessable. The solution also has the advantage of preventing the man-in-the-middle attack illustrated in Figure 4.

However, the downside in the EKE protocol is its inability to fully support two-factor one-time password authentication. This is because EKE requires that the password be used to encrypt the randomly generated key. The server is expected to have the same password to decrypt the generated key which may not be possible since most one-time password implementations (e.g., S/Key, RSA SecurID, VASCO Digipass) at the server-end only verify, and do not or cannot produce the expected

Figure 3. Interlock protocol to defeat man-in-the-middle attack

| Alice | | Mallory | | Bob |
|--------------------------------|---|-------------------------------------------------------------|---|--------------------------------|
| $g^a \text{ mod } n$ | → | | | |
| | ← | $g^m \text{ mod } n$ | | |
| | | $g^m \text{ mod } n$ | → | |
| | | | ← | $g^b \text{ mod } n$ |
| $k = g^{ma} \text{ mod } n$ | | $K = g^{am} \text{ mod } n$ $K' = g^{bm} \text{ mod } n$ | | $K' = g^{mb} \text{ mod } n$ |
| $E_k(\text{secret info}_{a1})$ | → | $E_k(??)$ | → | |
| | ← | $E_k(??)$ | ← | $E_k(\text{secret info}_{b1})$ |
| $E_k(\text{secret info}_{a2})$ | → | got secret info _a !! $E_k(??)$ | → | |
| | | | | REJECT |

Figure 4. Interlock protocol is unable to defeat man-in-the-middle attack for authentication

| Alice | | Mallory | | Bob |
|------------------------------|---|-------------------------------------------------------------|---|---------------------------------|
| $g^a \text{ mod } n$ | → | | | |
| | ← | $g^m \text{ mod } n$ | | |
| | | $g^m \text{ mod } n$ | → | |
| | | | ← | $g^b \text{ mod } n$ |
| $k = g^{ma} \text{ mod } n$ | | $k = g^{am} \text{ mod } n$ $k' = g^{bm} \text{ mod } n$ | | $K' = g^{mb} \text{ mod } n$ |
| $E_k(\text{auth info}_{a1})$ | → | | | |
| | ← | $E_k(??)$ | | |
| $E_k(\text{auth info}_{a2})$ | → | got auth info _a !! | | |
| | ← | Disconnect Alice $E_{k'}(\text{auth info}_{a1})$ | → | |
| | | | ← | $E_{k'}(\text{auth info}_{b1})$ |
| | | $E_k(\text{auth info}_{a2})$ | → | |
| | | | ← | $E_{k'}(\text{auth info}_{b2})$ |

one-time password (Haller, Metz, Nesser & Straw, 2005).

FURTHER OBSERVATIONS

Both the interlock protocol and EKE protocol have shown some potential in preventing man-in-the-middle attacks. However, by simply applying the protocols to Web-based transactions is not viable as both solutions rely heavily on both the client and server performing a stipulated set of operations as per required in the protocol. In the case of Web-based transactions, there is no way this can be ensured since the executable content on the user’s browser is determined by the hacker. Mallory could easily present a set of similar looking HTML content on the browser and Alice would be fooled to think that the actual cryptographic protocol was taking place, when in fact, the authentication information entered by Alice was presented directly to Mallory.

authentication information entered by Alice was presented directly to Mallory.

In Figure 5a, Alice is phished for the authentication information and the interlock/EKE protocol implemented by Bob cannot prevent Mallory from carrying out a man-in-the-middle attack.

To complete the last piece of the puzzle, we postulate the scenario that even if Alice and Bob authenticated each other correctly during the login process, Alice is still vulnerable to the man-in-the-middle attack as Mallory can easily hijack the Web session after the authentication has taken place.

This is because the HTTPS session encryption between the browser and the Web server is not correlated to the authentication exchange between the communicating parties. The authentication information entered by Alice played no part in deciding the ses-

Figure 5a. Phishing attack for authentication

| Alice | | Mallory | | Bob |
|-----------------|---|----------------------------------------------------------------------|---|--------------------------------|
| | ← | fake login Web page | | |
| enter auth info | → | | | |
| | | Perform protocol exchange with Bob using auth info provided by Alice | ↔ | Interlock/EKE protocol |
| | | | ← | Ok secret info _b |
| | ← | secret info _b | | |

Figure 5b. Session hijack after successful authentication

| Alice | | Mallory | | Bob |
|------------------------------------|---|--------------------------|---|--------------------------------|
| Perform protocol exchange with Bob | ↔ | pass-through proxy | ↔ | Interlock/EKE protocol |
| | | | ← | Ok secret info _b |
| | ← | secret info _b | | |

sion encryption key used for that session. In fact, the session encryption negotiation would have already taken place before Alice entered the authentication information. To achieve the attack, Mallory has to simply compromise the HTTPS session encryption by operating as a pass-through proxy, and hijack the session after the authentication has successfully taken place.

The challenge therefore is to find a Web-based authentication implementation that :

- prevents Alice from falling prey to Mallory's fake login Web page
- cannot be successfully operated by Mallory using a reverse-proxy or pass-through implementation

- is able to support two-factor authentication
- is still convenient to Alice for Web-based usage.

PROPOSED SOLUTION— TWO-FACTOR INTERLOCK AUTHENTICATION

Our proposed solution exploits the cryptographic foundation of the interlock protocol, with the advantages of two-factor authentication, and certain practical aspects of Web browser usage to arrive at a realistic solution in defeating active phishing. The EKE is not suitable for two-factor one-time password authentication as it relies on the knowledge of the password in order to derive the encryption key. This is not pos-

sible in the server-end for many one-time password systems.

We will first illustrate the proposed two-factor interlock authentication protocol (TIAP) before describing the solution. The idea is to split the two-factor authentication information into two portions, the static password (Pwd) and the one-time password (OTP), of which the OTP is first presented, before the Pwd. By doing so, we will be able to defend against Bellovin and Merritt's attack on the interlock protocol (Bellovin and Merritt, 1992). The proposed protocol is as follows in Figure 6:

- Steps 1 to 3 is unchanged from the standard Diffie-Hellman exponential key exchange protocol, and simply describe a means for both Alice and Bob to obtain a session encryption

key. Besides using the Diffie-Hellman exponential key exchange protocols, the use of other public key based protocols, such as RSA is also possible. The important requirement in the key exchange is that the session key must not be solely determined by only one party but by the combination of two parties.

- Step 4 to Step 7 is the interlock protocol where the one-time password is submitted by Alice to Bob, while the user information is sent from Bob to Alice. In Step 4, the one-time password (OTP) information is encrypted and only the first half of this information is sent to Bob. Alice will also indicate specific user information (such as the user name, salutation, last login time, and last failed attempt) that Bob has to

Figure 6. Two-factor interlock authentication protocol (TIAP)

| Step | Alice (user) | | Bob (server) | Remarks |
|------|----------------------------------------------------------|---|-------------------------------------------------------------|----------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| 1 | $g^a \text{ mod } n$ | → | | This is essentially a Diffie-Hellman exponential key exchange. |
| 2 | | ← | $g^b \text{ mod } n$ | |
| 3 | $k = g^{ba} \text{ mod } n$ | | $k = g^{ab} \text{ mod } n$ | |
| 4 | $E_k(\text{OTP info}_{a1})$ | → | | Doing the first Interlock protocol will defeat the man-in-the-middle attack as shown by Rivest and Shamir (1984). Since we are exchanging only the OTP (one-time-password) here, the danger of a replay attack is removed. |
| 5 | | ← | $E_k(\text{user info}_{b1})$ | |
| 6 | $E_k(\text{OTP info}_{a2})$ | → | | By enforcing a round-trip exchange of information both from Alice to Bob and from Alice to Bob, Bellovin and Merritt's attack (1992) on the interlock protocol will not succeed. |
| 7 | | ← | Check OTP info _a $E_k(\text{user info}_{b2})$ | |
| 8 | Check user info _b $E_k(\text{Pwd info}_a)$ | → | | |
| 9 | | ← | Check Pwd info _a Ok | |
| 10 | $E_k(\text{secret info}_a)$ | → | | |
| 11 | | ← | $E_k(\text{secret info}_b)$ | |

present to Alice. The reason why the one-time password is sent first, rather than the static password, is to prevent a replay attack by Mallory.

- In Step 5, Bob will encrypt the session-chosen user-specific information, and send the first half of the user information back to Alice.
- In Step 6, Alice will send the second half of the OTP information to Bob.
- In Step 7, Bob will first verify that the OTP information is correct, before sending the second half of the user information to Alice. Now as a follow up from Step 7 to Step 9, an additional round-trip of information from Alice to Bob has to take place. This is essential to break Bellovin and Merritt's attacks on the interlock protocol for authentication (Bellovin and Merritt, 1992).
- In Step 8, Alice will be prompted with the user information from Bob and asked to verify that the user information is correct before encrypting and sending the static password information to Bob.
- In Step 9, Bob will verify the static password information, and proceed with the session if verified correct.
- Now that the authentication protocol has established a common session key between Alice and Bob, sensitive information (e.g., transaction data) to be transferred between Alice and Bob should be encrypted using this session key.

We propose that the implementation of the Two-factor Interlock Authentication Protocol on Alice's side will be by means of a Java applet (or ActiveX) plug in embedded within the browser. It is important to use compiled plug ins, rather than script-

based content (e.g., Javascript), as current reverse-proxy implementations are unable to lift content displayed by the compiled plug ins. The Java applet plug in will be responsible for:

- negotiating the session key (Step 1 to 3)
- prompting the entry of the one-time password, and encrypting it (Step 4)
- transmitting the first half of the encrypted one-time password to Bob (Step 4)
- transmitting the second half of the encrypted one-time password to Bob (Step 6)
- decrypting the user information, and displaying it to Alice for verification (Step 8)
- prompting the entry of the static password and encrypting it (Step 8). This step could be enhanced with the use of EKE to strengthen the transmission security of the static password, but this is not related to defending against phishing attacks.
- transmitting the encrypted static password to Bob (Step 8)
- functioning as a cryptographic API to encrypt/decrypt sensitive information to be exchanged between Alice and Bob.

As for the server-side implementation, the authentication service operated by Bob has to be able to handle the TIAP exchange, and importantly be able to authenticate the one-time password and static password separately.

PROTOCOL ANALYSIS

We will analyze the proposed TIAP by simulating the possible hacking attacks to

be carried out by Mallory in the various steps. The places of attack are:

- Eavesdrop the entire conversation
- Collect Alice’s OTP information before masquerading as Alice to Bob
- Collect Bob’s user information before masquerading as Bob to Alice
- Send fake HTML content to Alice to prevent Alice from using the protocol
- Perform a session-hijack on the session after authentication has taken place

Hacking Attempt 1—Eavesdrop the Entire Conversation

We first assume that Mallory has been successful at compromising the session key (Steps 1 to 3).

In attempting to simply eavesdrop on the authentication flow, the hacking attempt by Mallory is thwarted by the Interlock pro-

ocol between Alice and Bob in Step 4b.

Hacking Attempt 2—Masquerade as Alice

In the second attempt, Mallory becomes smarter and attempts to collect Alice’s OTP information before opening a connection to Bob in Step 4b.

By deferring the communication with Bob, Mallory is able to first collect Alice’s OTP information and open a valid channel with Bob. However, this hacking attempt is again thwarted by the interlock protocol as Mallory is unable to present the user information to Alice in Step 8a.

Hacking Attempt 3—Masquerade as Bob

Now that Mallory has captured a specific set of Alice’s user information, he will attempt to masquerade as Bob to obtain Alice’s static password information before

Figure 7. Hacking attack 1—eavesdropping

| Step | Alice | | Mallory | | Bob |
|------|-----------------------------|---|-------------------------------------------------------------|---|-------------------------------------------------|
| 1 | $g^a \text{ mod } n$ | → | | | |
| 2a | | ← | $g^m \text{ mod } n$ | | |
| 2b | | | $g^m \text{ mod } n$ | → | |
| 2c | | | | ← | $g^b \text{ mod } n$ |
| 3 | $k = g^{ma} \text{ mod } n$ | | $k = g^{am} \text{ mod } n$ $k' = g^{bm} \text{ mod } n$ | | $k' = g^{mb} \text{ mod } n$ |
| 4a | $E_k(\text{OTP info}_{a1})$ | → | | | |
| 4b | | | $E_{k'}(??)$ | → | |
| 5a | | | | ← | $E_{k'}(\text{user info}_{b1})$ |
| 5b | | ← | $E_k(??)$ | | |
| 6a | $E_k(\text{OTP info}_{a2})$ | → | got OTP info _a !! | | |
| 6b | | | $E_{k'}(??)$ | → | Incorrect OTP !! no part 2 of user info sent |

Figure 8. Hacking attack 2—masquerade as Alice

| Step | Alice | | Mallory | | Bob |
|------|-----------------------------------|---|-------------------------------------------------------------|---|---------------------------------|
| 1 | $g^a \text{ mod } n$ | → | | | |
| 2a | | ← | $g^m \text{ mod } n$ | | |
| 2b | | | $g^m \text{ mod } n$ | → | |
| 2c | | | | ← | $g^b \text{ mod } n$ |
| 3 | $k = g^{ma} \text{ mod } n$ | | $k = g^{am} \text{ mod } n$ $k' = g^{bm} \text{ mod } n$ | | $k' = g^{mb} \text{ mod } n$ |
| 4a | $E_k(\text{OTP info}_{a1})$ | → | | | |
| 4b | | ← | $E_k(??)$ | | |
| 6a | $E_k(\text{OTP info}_{a2})$ | → | got OTP info _a !! | | |
| 6b | | | $E_{k'}(\text{OTP info}_{a1})$ | → | |
| 6c | | | | ← | $E_{k'}(\text{user info}_{b1})$ |
| 6d | | | $E_{k'}(\text{OTP info}_{a2})$ | → | |
| 7a | | | Got user info !! | ← | $E_{k'}(\text{user info}_{b2})$ |
| 8a | no user info, no Pwd info sent | ← | $E_k(??)$ | | |

Figure 9. Hacking attack 3—masquerade as Bob

| Step | Alice | | Mallory | | Bob |
|------------------------------------------------------------|------------------------------------------|---|-------------------------------------------------------------|---|------------------------------|
| Perform Hacking attempt 2 to get Alice's user information. | | | | | |
| 1 | $g^a \text{ mod } n$ | → | | | |
| 2a | | ← | $g^m \text{ mod } n$ | | |
| 2b | | | $g^m \text{ mod } n$ | → | |
| 2c | | | | ← | $g^b \text{ mod } n$ |
| 3 | $k = g^{ma} \text{ mod } n$ | | $k = g^{am} \text{ mod } n$ $k' = g^{bm} \text{ mod } n$ | | $k' = g^{mb} \text{ mod } n$ |
| 4a | $E_k(\text{OTP info}_{a1})$ | → | | | |
| 4b | | ← | $E_k(\text{user info}_{b1})$ | | |
| 6a | $E_k(\text{OTP info}_{a2})$ | → | got OTP info _a !! | | |
| 7a | | ← | $E_k(\text{user info}_{b2})$ | | |
| 8a | Incorrect user info, no Pwd info sent | | | | |

opening the connection to Bob. This attack is carried out by first performing the attack in Figure 8. When Alice reattempts to connect to Bob, Mallory will attempt to masquerade as Bob.

This attack would succeed if not for the requirement in Step 4 of the two-factor interlock protocol which requires that Alice indicates user-specific information (such as the user name, salutation, last login time, and last failed attempt) that Bob has to return. In Alice's reattempt to connect to Bob, the user information specified by Alice in Step 4a should be different from the first attempt, ensuring that Mallory is unable to present the correct user information back to Alice. Since there are so many different possibilities of user information to choose from, it would be very difficult for Mallory to phish Alice for all the possibilities

without Alice being suspicious. Naturally, this protocol requires Alice to be vigilant in checking for the user information.

Hacking Attempt 4—Fake Client Content

Mallory would have realized by now that the protocol is not possible to break. Instead, he deploys fake HTML content in Alice's browser in an attempt to circumvent the Interlock protocol.

This hacking attempt is thwarted by the use of Java Applet plug ins to carry out the protocol as well as to display the user information returned by Bob. Since current reverse-proxy technology is unable to capture whatever is displayed by an Applet in Step 7b, Mallory is unable to dynamically present the user information back to Alice and hence is prevented from

Figure 10. Hacking attack 4—fake client content

| Step | Alice | | Mallory | | Bob |
|------|--------------------------------|---|--------------------------------------------------------------|---|---------------------------------|
| | | ← | fake Web site | | |
| 1 | | → | | | |
| 1a | | | $g^m \text{ mod } n$ | → | |
| 2a | | | | ← | $g^b \text{ mod } n$ |
| 3 | | | $k' = g^{bm} \text{ mod } n$ | | $k' = g^{mb} \text{ mod } n$ |
| 4a | Alice enters OTP | → | | | |
| 6a | | → | got OTP info _a !! | | |
| 6b | | | $E_{k'}(\text{OTP info}_{a1})$ | → | |
| 6c | | | | ← | $E_{k'}(\text{user info}_{b1})$ |
| 6d | | | $E_{k'}(\text{OTP info}_{a2})$ | → | |
| 7a | | | Got user info !! | ← | $E_{k'}(\text{user info}_{b2})$ |
| 7b | | | Reverse proxy unable to extract user information from Applet | | |
| 8a | no user info, no Pwd info sent | ← | ?? | | |

succeeding in the hacking attempt. Not to forget that it is possible for Mallory to perform an off-line reverse-engineer on the Applet to allow the user information to be extractable in real-time, it is important for Bob to deploy the use of different applets, each varying in the Diffie-Hellman public keys as well as the way the two halves of the data is split and joined, for different authentication sessions.

Hacking Attempt 5—Session Hijack after Authentication

In the final hacking attempt, Mallory will allow the entire authentication flow to pass through before taking over the session.

This attack would have been easily carried out if not for the shared session key between Alice and Bob. While Mallory retains control of the underlying HTTPS session, he is unable to access or modify

information exchanged between Alice and Bob which is encrypted using the shared session key, thus effectively shutting Mallory out from the communication.

FUTURE RESEARCH

We have proposed and described the two-factor interlock authentication protocol which is an adaptation of the interlock protocol with two-factor authentication and deployment recommendations to suit the protocol for defeating active phishing in Web-based transactions. We have also comprehensively carried out hacking scenarios to attempt to break the protocol which seem to be able to withstand the attacks. We believe that the proposed approach can be used as part of the enterprise-wide antiphishing strategy.

At face value, the protocol is not entirely foolproof as the weakest point in

Figure 11. Hacking attack 5—session hijack after authentication

| Step | Alice | | Mallory | | Bob |
|---------------------|-----------------------------|---|----------------------------------------|---|------------------------------|
| 1 | $g^a \text{ mod } n$ | → | $g^a \text{ mod } n$ | → | |
| 2 | | ← | $g^b \text{ mod } n$ | ← | $g^b \text{ mod } n$ |
| 3 | $k = g^{ba} \text{ mod } n$ | | | | $k = g^{ab} \text{ mod } n$ |
| 4 | $E_k(\text{OTP info}_{a1})$ | → | $E_k(\text{OTP info}_{a1})$ | → | |
| 5 | | ← | $E_k(\text{user info}_{b1})$ | ← | $E_k(\text{user info}_{b1})$ |
| 6 | $E_k(\text{OTP info}_{a2})$ | → | $E_k(\text{OTP info}_{a2})$ | → | |
| 7 | | ← | $E_k(\text{user info}_{b2})$ | ← | $E_k(\text{user info}_{b2})$ |
| 8 | $E_k(\text{Pwd info})$ | → | $E_k(\text{Pwd info})$ | → | |
| 9 | | ← | Ok | ← | Ok |
| session is hijacked | | | | | |
| 10a | Info_a | → | Got Info_a Info_m | → | |
| 10b | | ← | Got Info_b Info_m | ← | Info_b |
| 10c | $E_k(\text{Info}_a)$ | → | ?? | | |
| 10d | | | ?? | ← | $E_k(\text{Info}_b)$ |

the protocol still relies on the vigilance of the end user to check for user information returned by the server in order to detect possible malicious activity. This article also did not cover how Trojan horses or viruses on the users' machine may affect the solution as we believe that this problem should be addressed through the use of antivirus software installed on the machine. Hence, in deploying this protocol, it is also important to incorporate a security-awareness program to constantly remind users on the proper usage of the Web site. Future research would be to deploy a reference implementation, most likely with an SSL-VPN, to understand other practical aspects (such as performance tuning and streamlining the protocol) of the TIAP solution before possibly claiming victory against the phishing attacks.

REFERENCES

- Bellovin, S. M. (2004). Spamming, phishing, authentication, and privacy. *Communications of the ACM*, 47(12), 144.
- Bellovin, S., & Merritt, M. (1992). *Encrypted key exchange: Password-based protocols secure against dictionary attacks*. Paper presented at the IEEE Symposium on Security and Privacy, Oakland, CA.
- Bellovin, S., & Merritt, M. (1994). An attack on the interlock protocol when used for authentication. *IEEE Transactions on Information Theory*, 40(1), 273-275.
- Diffie, W., & Hellman, M. E. (1976). New directions in cryptography. *IEEE Transactions on Information Theory*, IT-22, 644-654.
- Geer, D. (2005). Security technologies go phishing. *IEEE Computer*, 38(6), 1821.
- Goth, G. (2005). Phishing attacks rising, but dollar losses down. *IEEE Security and Privacy*, 3(1), 8.
- Haller, N., Metz, C., Nesser, P., & Straw, M. (2005). *A one-time password system*. Retrieved May 5, 2006, from <http://www.faqs.org/rfcs/rfc2289.html>.
- Herzberg, A. (2004). *Web spoofing and phishing attacks and their prevention*. Paper presented at the Fifth Mexican International Conference in Computer Science, Colima, Mexico.
- Inomata, A., Rahman, M., Okamoto, T., & Okamoto, E. (2005). *A novel mail filtering method against phishing*. Paper presented at the IEEE Pacific Rim Conference on Communications, Computers and Signal Processing, Victoria, B.C., Canada.
- Kirda, E., & Kruegel, C. (2005). *Protecting users against phishing attacks with anti-Phish*. Paper presented at the 29th Annual International on Computer Software and Applications Conference, Edinburgh, Scotland.
- Kirk, J. (2005). *Yahoo users get phished*. Retrieved May 6, 2006, from <http://www.pc-world.com/news/article/0,aid,122707,00.asp>.
- Liu, W., Huang, G., Liu, X., Deng, X., & Zhang, M. (2005). *Phishing web page detection*. Paper presented at the Eighth International Conference on Document Analysis and Recognition, Daejeon, Korea.
- Rivest, R., & Shamir, A. (1984). How to expose an eavesdropper. *Communications of the ACM*, 27(4), 393-394.
- Schneier, B. (2005). *The failure of two-factor authentication*. Retrieved May 6, 2006, from http://www.schneier.com/blob/archives/2005/03/the_failure_of.html.

ENDNOTES

- 1 Alice would use $(g^b \bmod n)^a \bmod n = g^{ab} \bmod n$, while Bob would use $(g^a \bmod n)^b \bmod n = g^{ab} \bmod n$
- 2 How the halves are derived depends on the encryption algorithm. In the case of a block cipher such as 3DES, the 2 halves will correspond to the first 4 bytes and last 4 bytes of each 8 byte encrypted block.

Xin Luo is an assistant professor of computer information systems in Department of Computer Information Systems of School of Business at Virginia State University, USA. He received his PhD in information systems from Mississippi State University in 2007. He has an undergraduate degree in International Business from Sichuan Normal University, China, an MBA from The University of Louisiana, and an MSIS in Information Systems from Mississippi State University. His research interests center around information security, E-commerce/M-commerce, and global IT adoption and management. He is the assistant editor of Journal of Internet Banking and Commerce. He has published numerous research papers and attended international & national conferences including Communications of the ACM, Information Systems Security, JIBC, AMCIS, DSI, IRMA, and etc.

Teik Guan is the chief security officer at Data Security Systems Solutions Pte Ltd in Singapore. He has many years of hands-on experience in designing and implementing data security solutions for many highly-sensitive projects including the Interbank RTGS Systems, Cheque Clearing Systems and several Internet banking and trading systems. Teik Guan is well-versed in the niche area of cryptographic security programming and integration, having developed numerous successful products such as CAs, smartcards (Javacard), HSMs, authentication servers, etc. Teik Guan holds a MSc from the National University of Singapore.