Fighting a bank spoofing attack: an out of page security channel

Michal Takács[*]

Slovak University of Technology
Faculty of Informatics and Information Technologies
Ilkovičova 3, 842 16 Bratislava, Slovakia
takacs@fiit.stuba.sk

Abstract. Internet crime is rising at a very high rate. As methods of mathematical security are being applied on many Internet communication channels previously left unprotected, the attackers are using another methods to fulfill their goals. Concurrent Internet sofware fails to deliver existing security to the everyday user. Every day, more spohisticated phishing and spoofing methods are announced.

This paper performs a case study – a web banking application spoofing attack. Details of the attack are analysed and explained. Reasons of existung security failure are discussed and a solution based on a security toolbar is proposed.

1 Introduction

In this paper we concern phishing and spoofing attacks on the users of web based banking applications. Email phishing is a form of digital identity theft where an adversary tricks an email recipient into revealing personal information [1].

The goal of these attacks is to trick the users into thinking that they are communicating with a real web application, but instead they are routed through attacker's interface [6].

The SSL [7] was designed to deal with the problems of an attacker web site pretending to be a real, well known web site, like a banking web application. For various reasons, SSL sometimes fails to deliver the security it was expected to [11].

When SSL is installed, the attackers may try the following:

[*] Supervisor: prof. Ing. Vladimír Vojtek, PhD., Institute of Applied Informatics, Faculty of Informatics and Information Technologies STU in Bratislava

IIT.SRC 2006, Bratislava, April 26, 2006, pp. 1-7.

- Use a phishing email [5] with no SSL connection. SSL to the bank will be made by the phishing web site.
- Use invalid SSL certificate that the user can be confused by. The user might click away the SSL warning and trust the indicated “secure connection”.
- Use valid SSL certificate for the phishing web site. The attacker can, as a part of detailed planning, buy a real certificate for his fraudulent site.
- Spoof DNS [3].

Analysis of user interface parts that have a negative impact on the SSL security is in [9].

Many sites even do not use SSL for private data transfer, because the complexness of the SSL certificate distribution scheme made it quite expensive for starting businesses. These sites are easy victims for phishers.

2 Case study: Bank account attack

This case study will concern banking web applications that use login/password for authentication and GRID card position to authorize money transfers. The authors know a minimal of two such applications – Tatra Banka and Slovenska Sporitelna currently available on the Slovak market. Probably many other banks use these authentication and authorization tools.

2.1 GRID card

The GRID card is a small card of size of a credit card. It usually contains a printed or embossed table of alphanumerical values, with marked column and row headings (similar to an Excel worksheet).

The card is issued by the bank and kept safe. Random values are requested upon something need to be authorized. This is done by generating random coordinates and requesting the value at these coordinates from the user. The server has a digital copy of the card and compares the value entered by the user. This brings the necessary randomness to the communication – even if it is eavesdropped, the attacker does not gain an automatic access next time.

2.2 The situation

We are interested in a situation, when the attacker can in some way read the communication between the user and the bank application. This can be established

with DNS spoofing, ARP spoofing [10] or via a phishing email. In the latter, the communication is routed through a false web site.

In all of these, SSL should have saved the day, but sometimes it fails because of the reasons mentioned before.

Common situations in the bank application user interface:

1. Login – the data sent contains the username and password, for some applications also grid card value. The attacker can successfully log in along with the user, routing the communication.

   Attacker can browse the whole application (sometimes, this can be suppressed using dynamic IDS [8]), while the user does not realize this.

2. Transfers - attacker can issue money transfer, but only when the user issues one, because he does not have a GRID card. The GRID value needs to be acquired from the user, for example in the way that the attacker presents the user with GRID coordinates server-generated for his undercover request. After fulfilling his transfer, he displays a message asking the real coordinates from the user, misleading the user into thinking that he made a typo. The user even finishes her own transfer request, leaving the fraud unnoticed.

3. Rest of the banking GUI - attacker can pretend that the bank requests arbitrary GRID position and acquire it to finish a transfer. Inexperienced user can fail to identify an unnecessary GRID request.

2.3 Reasons of failure

The user interface of an internet banking web application is by far too loosely defined to protect the user from man-in-the-middle attacks. This is a feature of any HTML based application. Any information that the server sends to the user (including a GRID card position indication) can be tampered with and changed according to the attacker needs.

As we have shown before, SSL and the like do not solve this problem, since current browsers display the certificate warnings in such a way that many users do not understand. However, more advanced users are well protected by SSL.

Thus, other secure channel between the user and the server application is needed to ensure no fraud actions can be taken upon an unexperienced user.

3 Our solution to this problem

As [4] mentions, the possibility to change every aspect of the web page layout by the attacker means that the secure communication channel can not operate via the web page.

There are many solutions for internet banking software that are not web based. We want to preserve the web GUI for its simplicity and flexibility and add a bit of security that can be later built in the web browsers, enhancing the security of banking web applications in general.

3.1 Trasfer signing

The server is computing the position on the GRID that is requested and it is questionable, whether this position is data based or not (the algorithms used are kept private).

Consider the following situation (later referred to as transfer signing):

1. User requests a money transfer.
2. Server provides the user with a form where basic transfer information is requested.
3. The filled form is sent to the server. This data will be denoted as D.
4. Server, based on the sent data D, computes the GRID position and requests the GRID data from the user. Let's denote this computation function GRID(X). The result of this function is the GRID position indicator (i.e., A3 or C1).
5. This way, the transfer is "signed" by the server and the transfer data cannot be tampered with. GRID(D) is sent to the user. User sends the appropritate GRID value to the server and the transaction is finished.

The man-in-the-middle attacker might try to compromise this transaction by performing these steps:

1. In the step 3, the form data is intercepted and changed in the way attacker wants – she can replace the target account number and amount, trasfering the victim's money to his account. The attacker's data sent to the server instead of D will be denoted as E.
2. The attacker sends E to the server, therefore GRID(E) position is requested and recorded for comparison by the server. The web page displays an input box and requests GRID(E). The user, unaware of the fact that GRID(D) != GRID(E), enters the value at GRID(E).
3. The attacker gets the value of required grid position and finished her malicious transfer.
4. The original transfer requested by the user, however, will not be finished. This could indicate a potential security threat to the user, but the attacker can simulate a typing error situation and the users are used to make small

mistakes. As the attacker detaches from the active participation on the communication and only retransmits it, the user finishes the original transfer. The fraud could be unnoticed for hours, or even days.

This attack clearly shows the weakness of security through server based security user interface. Such an interface can be, provided that the communication channel is compromised, altered on the way, unnoticed by the user.

3.2 Moving the signing to the client

One solution to this is to move part of this interface to the client. With SSL, this had already happened, in a way. The browser contains SSL certificate verification, which notifies the user when encountering a problem. Nevertheless, as studies show, internet fraud is on the rise [2], thanks to poorly defined warning and error reporting interface to technically weak users.

We propose that the GRID can be entered using a separate interface, a security toolbar. The *most important* part is that the server does not broadcast the GRID position, the function GRID(X) is implemented on the toolbar and on the server as well. This way, the GRID position cannot be spoofed and the steps 3 and 4 of the "transfer signing" from section 3.1 are secure.

The steps that need to be taken to create fully working solution include:

1. GRID(X) is defined. A simple solution includes hashing a concatenated string of the money transfer parameters and using a modulo of the resulting value to compute the GRID position. A formal specification for GRID(X) has to be established; it must be binary compatible among the client and the server, so both sides compute the same result.
2. The security toolbar with built-in GRID(X) is created. It should be created and distributed by the bank, probably only via offline channels, so no Trojan horses or viruses can be inserted.
3. The users should be instructed by the bank, that the only place they should enter the GRID value and the only source (and display) for the GRID coordinates is the toolbar. The attacker can try to spoof the toolbars and request the GRID values also on the page.

4 Conclusion

In this work, we have proposed a client-based way of creating a secure channel of money transfer authorization with GRID cards. A security toolbar should be created and the users should be instructed to use it when entering the GRID position.

The strengths and weaknesses are discussed in the last part of this work.

4.1 Strengths

- The attacker cannot compromise the additional secure channel that transports the GRID(X) value.
- Another out-of-page security feature is installed, moving the security on the client side; it can do more than in-page security.
- If the toolbar was created one-per-user, then, even if an attacker gathers all the values of the GRID card, there is still a probability of 1 / (GRID card cardinality) of a successful attack.

4.2 Weaknesses:

- Distribution of the toolbar must be kept very secure, backdoors could compromise the whole security.
- The algorithm for GRID(X) is hard coded in a publicly available toolbar. This way, any attacker actually knows the position of the GRID value for any transfer data. This can theoretically lead to increased GRID card theft. The card must remain secure at all costs appropriate – but this requirement is not new.
- As the attacker can predict GRID value requests, the gathering of the GRID values via eavesdropping becomes very valuable. We are still working on the improvement of this drawback.

4.3 Further work

Further research will be focused on improving the visibility of the toolbar – some forced human interaction should be introduced in order to raise user's perception. Also, the toolbar should be user-customizable, so the attacker does not benefit from eavesdropping at all.

Acknowledgement: This work was supported by Grant Agency of Slovak Republic grant No. VG1/3103/06.

References

[1] Adida, B., S. Hohenberger, et al. (2005). Separable Identity-Based Ring Signatures: Theoretical Foundations For Fighting Phishing Attacks. In submission.

[2] Anti Phishing Working Group (2006). Phishing Activity Trends Report, December, 2005, Last access February 2006 from http://www.antiphishing.org/reports/apwg_report_DEC2005_FINAL.pdf

[3] Bellovin, S. M. (1995). Using the Domain Name System for System Break-ins. Fifth Usenix UNIX Security Symposium.

[4] Dhamija, R. and J. D. Tygar (2005). The Battle Against Phishing: Dynamic Security Skins. Symposium on Usable Privacy and Security (SOUPS) 2005

[5] Christine E. Drake, J. J. O., and Eugene J. Koontz. (2004). Anatomy of a Phishing Email. Conference on Email and Anti Spam (CEAS).

[6] Edward W. Felten, D. B., Drew Dean, and Dan S. Wallach. (1997). Web Spoofing: An Internet Con Game, Princeton University Technical Report 540-96 (revised). Lst access February 2006, from http://www.cs.princeton.edu/sip/pub/spoofing.html.

[7] Freier, A. O., P. Karlton, et al. (1996). The SSL Protocol Version 3.0, work in progress. Last access February 2006, from http://tools.ietf.org/wg/tls/draft-ietf-tls-ssl-version3/draft-ietf-tls-ssl-version3-00.txt.

[8] Takács, M. (2004). An intelligent interaface to secure web applications. In proc. of DATAKON 2004, Brno, Czech republic. Pages: 311-320. Slovak language.

[9] Takács, M. (2006). Report on means and methods of modern Internet security, pre-doctoral thesis, Slovak language, Faculty of Informatics and Information Technologies, Slovak Technical University.

[10] Wagner, R. (2001). Address Resolution Protocol Spoofing and Man-in-the-Middle Attacks.

[11] Y. Yuan, E. Y., and S. Smith. (2002). Web Spoofing Revisited: SSL and Beyond, Technical Report TR2002-417. Last access February 2006, from http://www.cs.dartmouth.edu/~pkilab/papers/tr417.pdf.