How to Attack Two-Factor Authentication Internet Banking

Manal Adham[1], Amir Azodi[1,3], Yvo Desmedt[2,1], and Ioannis Karaolis[1]

[1] University College London, UK
[2] The University of Texas at Dallas, USA
[3] Hasso Plattner Institute, Germany

Abstract. Cyber-criminals have benefited from on-line banking (OB), regardless of the extensive research on financial cyber-security. To better be prepared for what the future might bring, we try to predict how hacking tools might evolve. We briefly survey the state-of-the-art tools developed by black-hat hackers and conclude that they could be automated dramatically. To demonstrate the feasibility of our predictions and prove that many two-factor authentication schemes can be bypassed, we developed three browser rootkits which perform the automated attack on the *client's computer*. Also, in some banks attempt to be regarded as user-friendly, security has been downgraded, making them vulnerable to exploitation. To wrap up, we discuss the measures needed to avoid the next generation of attacks.

1 Introduction

A lot of research has been done on the cyber-security of financial transactions, both from a cryptographic as well as from a computer security viewpoint. Due to work like [7], wholesale transactions are now relatively secure. The challenge today is to secure the PC-based transactions. We see that identity theft has been for more than a decade the most frequent white collar crime. We know that attacks against US and UK Internet banking systems have been quite successful [16]. Tools like the Zeus malware kit [15] and the SilentBanker Trojan [19], have been helping criminals perform fraudulent on-line bank transactions.

Since research should be ahead of hackers, the main goal of this paper is to predict the next generation of OB attacks and how they may worsen the already questionable security. We survey the tools hackers use and see that, they are not fully automated, requiring an involvement of humans to perform the fraudulent transactions. To demonstrate the feasibility of a fully automated attack, three advanced browser rootkits (for Firefox Browsers) were developed, targeting the Internet banking systems of NatWest, HSBC and Bank of Cyprus. An attack against Barclays' system is also proposed. Security of these major banks is largely dependent on the commonly used two-factor authentication schemes. By successfully circumventing and exploiting the weaknesses of such systems, we hope to raise awareness on how dangerous the current situation is.

Aside from security in the absolute sense, we consider auxiliary factors like usability. This is a critical factor that can drive a bank to lower its security threshold in order to satisfy its customers' desires (*e.g.*, Barclays).

2 Background

2.1 Attack Strategies and Techniques

We distinguish between three main Internet banking attack vectors that can be used alone or in combination. Firstly, a *Credential Stealing* attack, is where fraudsters try to gather users' credentials, either with the use of a malicious software or through phishing [8]. Secondly, a *Channel Breaking* attack, involves intercepting the communication between the client side and the banking server, by masquerading as the server to the client and vice versa [18]. Finally, a *Content Manipulation* (also called man-in-the-browser (MiTB) [14]) attack, takes place in the application layer between the user and the browser. The adversary is granted with privileges to read, write, change and delete browser's data "on the fly", whilst the legitimate user is seamlessly unaware.

A *Browser Rootkit*, is a content manipulation technique, which is capable of completely changing the browser's display and behavior [4]. It is basically a malicious *browser extension*, which are used to extend and/or customize the browser's functionality [1]. To control unauthorized installation of browser extensions, modern browsers like Firefox, have employed numerous but weak security measures, which can be easily bypassed [4].

2.2 Related Work and Attacks

A Firefox malicious extension called BROWSERSPY was developed, which once installed on the victim's machine, begins collecting personal data that is later sent to the remote attacker [20].

Zeus [15], *SilentBanker* [19] and *URLZone1* [2], are infamous Trojans, which have been successfully used against on-line banking systems (OBS) to steal millions of dollars [12, 16]. They are primarily used to steal login credentials and card numbers with their security codes, but can also change transaction details on the fly. This is commonly achieved by adding form fields at the browser level. The stolen data is sent to a command

and control server (C&C server) controlled by the remote attacker, who is then able to *manually* connect to the victims on-line banking accounts.

An attack against Helios 2.0 e-voting system is described in [4]. The attack exploits Adobe Acrobat/Reader vulnerabilities to install a browser rootkit on the voters' machines. This rootkit changes the selected legitimate candidate vote to the one chosen by the extension's author, and also misleads the voters to believe they have voted for their desired candidate.

In [9], an hPIN/hTAN device is proposed, which concentrates on the trade-off between security and usability. The central concept of the design is to make it simple.

3 Authentication Systems Used in Internet Banking

Currently, several banks implement an additional layer of secondary authentication, required for monetary transactions. Four common classes of authentication schemes are usually employed.

One-Time-Password (OTP) schemes, generate passwords using pseudorandom generators or mathematical algorithms in order *to authenticate the users*. However, they offer no security against man-in-the-middle (MiTM) or MiTB attacks [9]. *Bank of Cyprus* uses the DigiPass GO3 token for authenticating its customers' monetary transactions [21].

In *Full Transaction Authentication* schemes, the OTP generated *to authenticate the transaction* is cryptographically bound to the transaction data. For example, *Barclays'* PINsentry device, which adopts the CAP protocol, implements this scheme when in 'Sign mode" [17]. However, since August 2011, the implementation of PINsentry has been relaxed and *favors usability* rather than security. Specifically, account holders no longer need to use PINsentry for authenticating transactions when the payee is in their beneficiary list. It is evident that pressures from unhappy customers, who are unaware of security issues, have influenced Barclays to *downgrade its security* [5, 13].

Partial Transaction Authentication is a "relaxed" form of full transaction authentication, where only part of the transaction data are required during authentication. Example implementations of these systems include *NatWest* card reader (challenge/response mode) and *HSBC* security key. Here the cryptograms generated to authenticate transactions are based only on the last four digits of the payees' account numbers [17].

Finally, some OBSs do not require authentication of monetary transactions (*i.e., No Secondary Authentication* is required). This situation is true for NatWest and HSBC customers, who can perform payment transactions to payees who have previously been authenticated and to governmental/financial institutions deemed trustworthy by the banks.

4 Automation of the Attack

Current black-hat hackers' software requires manual intervention to perform fraudulent transactions, limiting the damage that can be caused. In the future, hackers may adapt their tools to take advantage of the weaknesses in modern OBSs and fully automate their attack. In order to demonstrate the current shortcomings of OBSs, we have implemented three browser rootkits that undermine the OB security of HSBC, NatWest and Bank of Cyprus. An attack against Barclay's OBS is also proposed.

4.1 Automatic Fraudulent Transaction

Once the browser rootkit is installed on the victims' machines, it passively monitors their browsing activities, and activates once they successfully log onto the targeted OB.

In order to achieve successful fraudulent transactions, some conditions must first be met. In the case of *HSBC* and *NatWest*, the last 4 digits of the beneficiary's account number must match the respective last 4 digits of an account controlled by the attacker (as this paper is a proof of concept, the attacker's account numbers are stored in the browser rootkits. To counter traceability, different account numbers, which can be mule accounts, can be fetched from different locations). In the case of *Bank of Cyprus*, a freshly generated OTP must be typed. When the necessary condition is met, the respective browser rootkit can successfully execute a fraudulent transaction by changing the transaction data "on the fly" (*i.e.*, the beneficiary's account number and amount), whilst displaying what the user expects to see (*i.e.*, the intended by the user transaction data). In order to keep the attacks undetectable for a longer period, a small percentage of the holdings in an account is taken each time.It is more likely that the victim will disregard a few missing dollars compared to a few hundred missing dollars.

At the time this study began, *Barclays'* OBS required a fresh full authenticating signature from PINsentry for every monetary transaction, thus no browser rootkit was feasible on its own to successfully attack the given system (full transaction authentication for every transaction beats browser rootkit attacks). However, the fact that account numbers in the user's beneficiary list need no signature, creates a vulnerability. Specifically, if the attacker is successful in tricking the victim to purchase or buy a fake service (*e.g.*, through scareware [11]), then the account number used by the attacker will be listed in the victim's beneficiary list. The browser rootkit will then be capable to perform fraudulent transactions to the attacker's account number, since money transfers to that account will require no signature! Similarly with *NatWest and HSBC*, money transfers to a host of businesses (deemed trustworthy) do not require a transaction verification step. In this case, the browser rootkit can be instructed to perform funds transfers to these hosts, hidden from the user's view. Although there is no direct financial gain for the attacker, this remains a vulnerability of this specific OBS and other similar.

4.2 Hiding the Attacks

The developed browser rootkits cover up all traces of their activities. Specifically, they store the information pertaining to the fraudulent transactions in browser cookies and on the victim's hard drive (under Firefox's "Profiles" directory). All fraudulent activity is successfully hidden from the victim by first, having the browser displaying the expected by the victim account balance rather than the actual one (which obviously has less money since the browser rootkit fraudulently transfered money from it), and secondly, by removing all fraudulent activity from the victims' statements. This is achieved by exploiting DOM elements of the HTML page. The fraudulent activity is kept hidden from the user no matter when and how many times the user logs out and back in again. As long the malicious extension is installed, the attack remains hidden.

4.3 What is New

The biggest advantage of the developed browser rootkits is that they are fully automated. Once they are successfully installed, they are capable of performing fraudulent transactions and covering their traces at the same time, without any further human intervention or instructions from a remote C&C server, as in the case of Zeus and the like Trojans [2, 15, 19]. This is feasible since the attacks are against *fixed* banks, with the browser rookits explicitly designed to attack the given OB systems of the respective targeting banks. In addition, the adversarial bank account numbers are stored in the browser rootkits code.

Additionally, no theft of banking information is necessary. Our browser rootkits are clever enough to remain hidden and submit when successful fraudulent transactions are possible.

Finally, in contrast with Trojans of black-hat hackers that create files and modify the registry keys in the victims' machines, our developed browser rootkits are simple browser extensions written in a few hundred lines of code. Browser rootkits are really hard detected by any antivirus program, since once installed they become a part of the browser. In contrast, OS rootkits and Trojans are much easier to detect with an updated antivirus or an Intrusion Detection System.

5 Defenses

It is possible to implement security measures that protect client machines from browser rootkits. The simplest defense, is the adoption of security devices that enforce *all monetary transactions to be fully authenticated*, including payments to governmental institutes or other public services (browser rootkits can be instructed to just empty the victim's balance (*e.g.*, HSBC attack)).This means that at least the whole account number and the amount must be signed for each transaction. No trustee list should exist as this is exploitable as demonstrated in this paper. Furthermore, it is crucial that these devices are user-friendly, since customer dissatisfaction can lead banks to relax their security and in effect make their systems vulnerable.

The approach we propose is to adopt the Glass [6] approach and advance it. As suggested by Glass, the need for a display and a keyboard is of vital importance. Our device has a USB connection that powers the device and establishes a secure communication channel with the bank. This can be realized in a number of ways (*e.g.*, VPN, SSL/TLS). We also include two buttons to accept or reject any process. The process of Logging into the OBS requires the use of the device. The user powers the device via USB, which in turn asks for the user's PIN. If correct, the device will activate and attempt to establish a secure communication channel with the OBS (*e.g.*, a VPN Connection). In order to make any transactions, the user will then have to enter the details of his desired transaction into the browser and submit them to the bank's server. The bank will sign the transaction based on the details it has received, and encrypt both the transaction details and its signature. The ciphertext with its MAC are then sent to the security device through the VPN connection. The device checks the integrity of the message and decrypts the ciphertext. The device will then display the recipient's account number and the amount. If confirmed by the user, the device will relay this confirmation through the VPN connection to the OBS and commit the transaction.

6 Future Work

Real deployments of two-factor authentication schemes suffer from security issues as a result of bad design, underestimating client-side attacks and lack of usability. An ideal scheme must accommodate for the trade-off between usability and security. Clearly, new ideas that focus on security with minimal user interaction are very important. Given that a secure element (*i.e.*, something the user has) has become essential to reach the standards of modern secure authentication schemes, the question of whether we can get rid of them may no longer stand. However, whether a universal security device for all OBSs could be deployed still remains, and depends on the willingness, effort and cooperation of the financial institutions. In addition, as the use of smartphones has transformed our daily lives, it is possible to use mobile phones to generate transaction authentication signatures. Mobile phones are in fact used as a second authentication, but the question is whether they can improve the current processes. Finally, some advances that could be addressed in a future work may be the use of Out-of-Band (OOB) communication channels [3]. In such a setting, the device is able to communicate with the authenticating servers through the secure OOB channel and display the information for confirmation.

7 Conclusions and Reflections

Two-factor authentication (excluding full transaction verification) is still inadequate to deal with browser rootkit attacks. Although the original user-unfriendly approach of Barclays shows that if criminals would automate their attacks, certain banks are ready to roll out their modifications and annul most of the attacks, the hardware/software used by most banks though, as HSBC, NatWest and Bank of Cyprus, may not allow them to switch quickly. We finally observe that full transaction verification may not fully address all security concerns. The information displayed on the PC including; account numbers, name, balance and transaction details, do *not* remain private! Indeed, a browser rootkit can leak all this information to an attacker who could use it to physically target rich users, use identity theft techniques, etc. To deal with this problem banks need to carefully consider what they display on the browser. Finally, this study has multiple precautions to ensure that no malware was released in the wild. No bank servers were violated. A longer version of this paper is available in [10].

Acknowledgments. We would like to thank the referees for their feedback and the shepherd of the paper Ahmad-Reza Sadeghi for his effort.

References

1. Mozilla Developer Centre. Extensions. https://developer.mozilla.org/en-US/docs.
2. D. Chechik. Malware Analysis Trojan Banker URLZone/Bebloh, September 2009. http://goo.gl/z7YSV.
3. D. Chou. Strong User Authentication on the Web, August 2008. http://goo.gl/6xbky.
4. S. Estehghari and Y. Desmedt. Exploiting the client vulnerabilities in internet E-voting systems: hacking Helios 2.0 as an example. In *Proceedings of the 2010 international conference on Electronic voting technology/workshop on trustworthy elections*, EVT/WOTE'10, pages 1–9, 2010.
5. PistonHeads Web Forum. Barclays PINsentry, what a dumb POS, 2008. http://goo.gl/wRoJV.
6. A. S. Glass. Could the Smart Card Be Dumb. In *Abstract of Papers, Eurocrypt 86*, pages 1.5A–1.5B, May 20–22, 1986. ISBN 91–7870–077–9, Linköping, Sweden.
7. F. M. B. Greenlee. Requirements for key management protocols in the wholesale financial services industry. *IEEE Communications Magazine*, 23(9):22–28, September 1985.
8. M. Jakobsson. Modeling and Preventing Phishing Attacks. In *Financial Cryptography.* Springer Verlag, 2005.
9. S. Li, A.-R. Sadeghi, S. Heisrath, R. Schmitz, and J. Ahmad. hPIN/hTAN: A Lightweight and Low-Cost e-Banking Solution against Untrusted Computers. In *Financial Cryptography and Data Security*, LNCS, pages 235–249. 2012.
10. M. Adham, A. Azodi, Y. Desmedt and I. Karaolis. How To Attack Two-Factor Authentication Internet Banking. 2013. http://goo.gl/YsA6j.
11. Microsoft Safety & Security Centre. Watch out for fake virus alerts. http://goo.gl/YEZMT.
12. E. Mills. Banking Trojan steals money from under your nose, September 2009. http://goo.gl/tuDfJ.
13. moneysavingexpert Web Forum. Stupid Barclays PINsentry Thingymajic. Can I Log Into My Online Account Without It?, 2010. http://goo.gl/eqaey.
14. RSA White Paper. Making Sense of Man-in-the-browser Attacks: Threat Analysis and Mitigation for Financial Institutions, 2010. http://goo.gl/NRcez.
15. S. Ragan. Overview: Inside the Zeus Trojan's source code, May 2011. http://goo.gl/nsvpG.
16. RiskAnalytics LLC. $70 Million Stolen From U.S. Banks With Zeus Trojan, October 2010. http://goo.gl/XkSgq.
17. S. J. M. Saar Drimer and R. Anderson. Optimised to Fail: Card Readers for Online Banking. *Financial Cryptography and Data Security*, February 2009.
18. B. Schneier. Two-Factor Authentication: Too Little, Too Late. *Commun. ACM*, 48(4):136–, April 2005.
19. Symantec. Banking in Silence, June 2009. http://goo.gl/aj61F.
20. M. Ter Louw, J. Lim, and V. Venkatakrishnan. Extensible Web Browser Security. In *Detection of Intrusions and Malware, and Vulnerability Assessment*, volume 4579 of *LNCS*, pages 1–19. Springer, 2007.
21. VASCO. DIGIPASS GO 3, August 2012. http://goo.gl/EmLFy.