Security of Online Banking Systems.

Technical report: Security of Online Banking Systems

Sven Kiljan[a,b,c,*], Koen Simoens[d], Danny De Cock[e], Marko van Eekelen[a,c], Harald Vranken[a,c]

[a] *Open Universiteit, Faculty of Management, Science and Technology, Postbus 2960, 6401DL Heerlen, The Netherlands*
[b] *NHL Hogeschool, Economy & Management - Lectureship Cybersafety, Postbus 1080, 8900CB Leeuwarden, The Netherlands*
[c] *Radboud University Nijmegen, Faculty of Science - Digital Security, Postbus 9010, 6500GL Nijmegen, The Netherlands*
[d] *Verizon Enterprise Solutions, Culliganlaan 2E, 1831 Diegem, Belgium*
[e] *KU Leuven, Department of Electrical Engineering - Computer Security and Industrial Cryptography, Kasteelpark Arenberg 10, bus 2452, B-3001 Leuven-Heverlee, Belgium*

Abstract

Technology keeps evolving to better support our everyday life. This also includes technology to manage finances. Online banking allows customers to access banking services not only from computers at home or work, but also almost anywhere using mobile devices. In the last few years, mobile devices have seen a great increase in adoption by banks and their customers for online banking due to their capabilities, affordability and usability. This report discusses the security of today's online banking systems. Our focus is on both online banking using home and office computers, and mobile banking using devices such as smartphones and tablets. We compare our findings with a similar examination of a decade ago and present an overview of security issues in online and mobile banking that exist today. The most important trends that we notice is that banks do improve security only after improvements in functionality are made, and that efforts so far did not solve the larger underlying issues.

Keywords: online banking, mobile banking, WWW security, mobile security, multi-factor authentication

*Corresponding author. *Telephone number:* +31 24 3652672
Email address: s.kiljan@cs.ru.nl (Sven Kiljan)

1. Introduction

Online banking systems have become quite popular in the last ten years. Customers from an online bank can manage their accounts with their own electronic devices as long as an Internet connection is available. Activities such as checking the balance of an account and the initiation of money transactions do not require human assistance from a clerk at the bank. Managing a bank account is not limited anymore to office hours.

Neither time nor location is an issue. Online banking systems are not limited to personal computers for customer interaction. Mobile devices such as smartphones and tablets allow customers to conduct banking activities 'on-the-go'. Mobile browsers and dedicated standalone applications allow people to use the Internet to connect to the bank's systems in a similar way to online banking with a PC.

1.1. About this document

This technical report is a result of our worldwide survey of security aspects in online banking. Its main focus is on our detailed findings and background descriptions. A more condensed version of this report is the basis for our journal article.

The target group for this report includes researchers with specific interests in security aspects that are practiced today. Examples include the use of SSL/TLS and (combinations of) authentication methods. Those who are only interested in developments of online banking security can also read the journal article.

1.2. Problem statement

We define two problems in online banking to which the contents of this report apply. The problem that is tangible and measurable is the financial fraud problem in online banking. Customers and banks lose money through illegitimate transactions. A less tangible and more difficult to quantify problem is the loss of privacy. Attackers that gain access to bank accounts have access to sensitive and incriminating information, such as account balances, transaction histories and information about debt. More information on how the identified issues in online banking contribute to the stated problems is given in Section 4.4 on page 47.

1.3. Scope of the report

This report discusses security technology and security usability in today's online banking systems. As a continuation of the work of Claessens et al. [1], the main focus will be on online and mobile banking.

An extensive worldwide survey of today's online banking systems provides the base of our work. We applied a scope similar to the work of Claessens et al. with a larger sample size by surveying available online banking systems all over the globe and the possible methods of interaction that are available to customers. The perspective of this report is from the side of the customer when dealing with online banking. Only publicly available information was used in our survey.

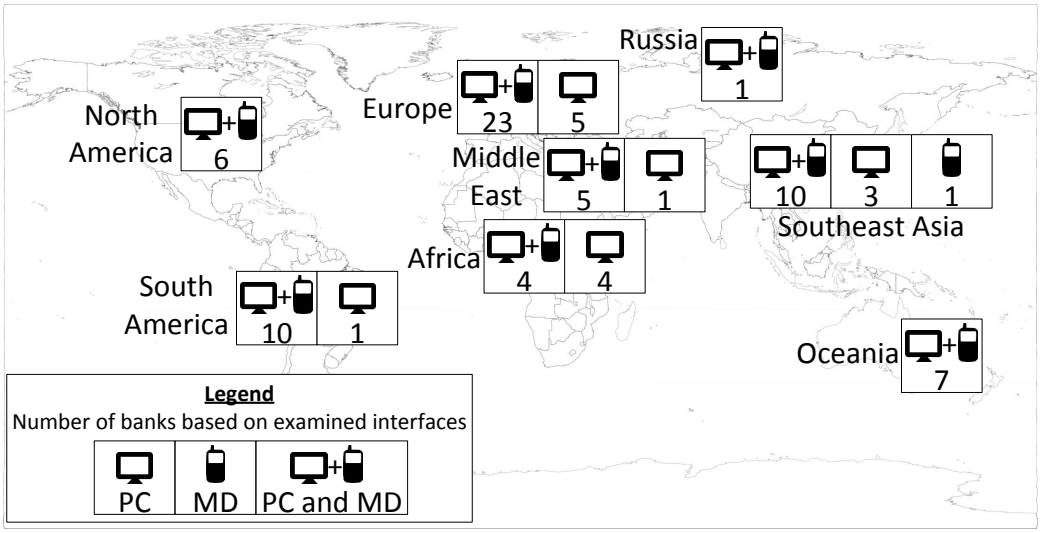

Figure 1: Overview of worldwide surveyed banks.

81 banks that provide online payment services were examined. See Figure 1 for an overview of the global distribution between North and South America, Europe, the Middle East, Africa, Russia, East-Asia and Oceania. Samples were collected by searching for banks on the Internet. Search criteria were based on global representation and type of bank. Our focus was on banks that provide account and payment services to consumers and small businesses. Data was collected between March and September in 2013. If a specific geographical area was targeted, individual banks were selected from the pool of banks that provide payment services. Several banks were included

based on economical size if this information was available. If this information was used, other random banks from the same region were also added. We explicitly added the larger banks where possible in the assumption that they have the most assets to protect (such as the number of customers or the amount of money) and are therefore a desired target. By including them in addition to random (smaller) banks, we think that our sample set is representative for online banking systems aimed at consumers and small businesses for the time period in which we conducted our research. In regions where the size of banks was not obtainable, we used a random selection. Language barriers were overcome by colleagues who translated critical information from documentation that we provided and by automated translation tools.

Due to technical, security and usability differences between the different types of customer devices, a distinction is made throughout this report between the use of personal computers (PCs) and mobile devices (MDs) by online banking customers. Of the banks in our set, 80 are examined on how they facilitate the use of the customer's PC.[1] The same was done for MDs with 67 banks in our set.[2] Figure 1 makes a distinction between banks that were examined for the services they provide for both PCs and MDs, and for banks that were only examined for one of these types of devices.

1.4. Outline of the report

The rest of the report after the introduction is organized as follows. In Section 2, we will give an overview of different aspects concerning online banking today and offset it to the developments ten years ago as described by Claessens et al. in [1]. This section first summarizes how popular online banking is and gives an indication about the increase of fraud. The architecture of today's online banking is explained, including applied authentication methods for online banking on both PCs and MDs. A focus is given on the applied authentication factors. These can be knowledge ('something you know'), possession ('something you have') and existence ('something you are').

[1] A single bank could not be examined because authentication options on their site were not visible unless a customer number was entered. Also, public documentation about the authentication process was not found.

[2] The other 14 banks either do not provide mobile banking services or we could not find information about it.

Section 3 continues with an overview of attack vectors applied against online banking. These vectors either misuse or work around the described authentication methods mentioned in the previous section. After this, an analysis is given in Section 4 of the underlying security issues that provide the largest impact on the financial fraud and privacy problems in online banking.

Section 5 contains our concluding remarks.

1.5. Our contribution

Our work gives an indication of the state of online banking worldwide and its current vulnerable aspects. Conclusions about the evolution of functionality versus security in online financial services are made by comparing the work of Claessens et al. in [1] with our work. Further research can use the defined security issues and problems in online banking as a first step towards improvement.

2. Today's online banking

In 2002, Claessens et al. described in [1] the security of online banking systems at the time. This section does the same, but starts with an overview of use and fraud statistics of online banking to give an indication about the rapid adoption rate and the amount of money lost due to security issues. After this overview, the focus will be on the architecture and several security aspects. Where appropriate, a comparison will be made between 2002 and 2013.

2.1. Use and fraud

The adoption of online banking has globally increased at a rapid pace. Where in the European Union (EU) online banking adoption was 19% of individuals aged 16 to 74 in 2005, it has increased to 37% in 2011.[3] In the United States, the number of bank customers that use online banking at the top 11 banks was 34.2 million in the first quarter of 2005 and 65.2 million in the same quarter of 2011.[4]

This relatively new and popular way of personal banking brings new approaches to commit fraud. Online banking fraud concerns the illegitimate

[3]Eurostat - Individuals using the Internet for Internet banking in the EU: `http://epp.eurostat.ec.europa.eu/tgm/refreshTableAction.do;?pcode=`

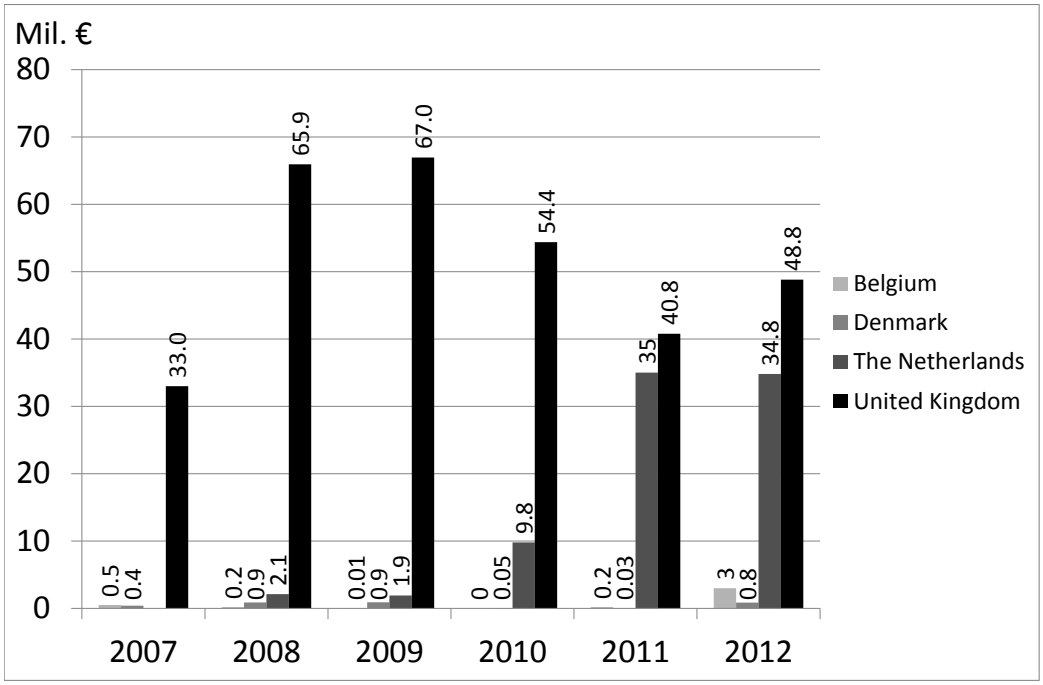

Figure 2: Known loss by banks due to online banking fraud in different countries (in millions of euro).

loss of financial assets through electronic channels that are normally used for legitimate transactions. See Figure 2 for an indication of lost financial assets by banks in specific countries in the past few years.[5,6] Amounts are in millions of euro. Conversion from Danish kroner and pound sterling to euro is based on the average conversion rate for each year.[7]

---

tin00099&language=en

[4]comScore, 2011 State of Online and Mobile Banking: http://theexchangenetwork.ca/upload/docs/comScore%202011%20State%20of%20Online%20and%20Mobile%20Banking.pdf

[5]Fraud damage for banks from online banking in 2007-2012 (Belgium, The Netherlands, United Kingdom and Denmark): ://www.safeinternetbanking.be/nl/cijfers/fraudegevallen

[6]Fraud damage for banks from online banking in 2008 (The Netherlands): https://www.security.nl/posting/31647

[7]OANDA - Average Exchange Rates: http://www.oanda.com/currency/average

2.2. Technical definitions

As discussed in Section 1.3, a distinction between PCs and MDs is made due to the large differences between both types of devices. To support this distinction, we give a definition of both types.

The line between what qualifies as a PC and what as an MD is starting to blur due to the increased functionality that MDs possess. Some tasks that required a desktop computer or laptop in the past can now be done on-the-go with tablets or smartphones. But there is more than functionality that defines the difference between PCs and MDs. For the terms used in this report a distinction is made between both types of devices based on their technical traits. This distinction is essential to understand what banks and customers can and cannot do with each type of device, and which security options are available.

- PCs are traditionally open platforms. Owners are unhindered in installing and using software. These are characterized by 'desktop' operating systems such as Linux(-based distributions), Apple Mac OS X and Microsoft Windows. Most desktop computers and notebooks belong in this category.

- Different types of MDs evolved from mobile phones. The focus on MDs in this report will be on smartphones and tablets.

  Smartphones allow users to install applications ('apps') that integrate with the mobile operating system to allow local execution of code. This offers a friendlier user experience and a better fit with the hardware compared to remote web-based applications that have to be accessed through a mobile browser.

  From smartphones, tablets were born. Although they differ from a physical perspective in that they are larger than smartphones, the underlying software architecture is often similar.

  The most popular mobile operating system between October 2012 and October 2013 was Google's Android, followed by Apple's iOS.[8] Other, less used mobile operating systems include Symbian by Accenture (on behalf of Nokia), BlackBerry OS by Research in Motion and Windows

[8]Used mobile operating systems between October 2012 and October 2013 by StatCounter: `http://gs.statcounter.com/#mobile_os-ww-monthly-201210-201310`

Phone by Microsoft. Android and iOS are used on both smartphones and tablets, while the other mobile operating systems are only used on smartphones.

When compared to traditional ('desktop') operating systems used by personal computers, mobile operating systems offer a more strict software environment. Most mobile operating systems allow the installation of applications only through specific sources. Applications also need to request permission from the user to access restricted information. Examples of this include the geographical location of the device and access to the user's address book.

Of course, there are more differences between both types of devices. For the distinction between the terms 'PC' and 'MD' in this report, the listed definitions are used. Other differences that apply in most cases and that have a significant impact will be mentioned where relevant. An example are the data input methods available to the user. Where MDs in most cases rely solely on the use of a touchscreen, PCs rely in most cases on the combination of a keyboard and mouse. This has a significant influence on the ability to enter complex passwords (lower and upper case letters, numbers and special characters), which is hampered when a touchscreen is used with virtual keyboards.

2.3. Architecture

This subsection gives an overview of the technical architecture that is provided and used by users, banks and other involved parties. Claessens et al. made a distinction between Internet architecture and WAP architecture, which can be seen as the situation for 2002 in Figure 3 and Figure 4. Since the capabilities of MDs are not limited anymore to mobile-only technologies, a distinction based solely on different underlying technologies would be cumbersome. Instead, we now recognize different architectures based on types of devices used by customers: PCs and MDs. This is significant for online banking due to the differences in software architecture and physical capabilities. These differences limit what a bank can and cannot offer on each type of device in both functionality and security.

2.3.1. Personal computers

Figure 3 illustrates the basic architecture of the Internet electronic banking system using a PC. Two parties are involved: the customer and the bank.

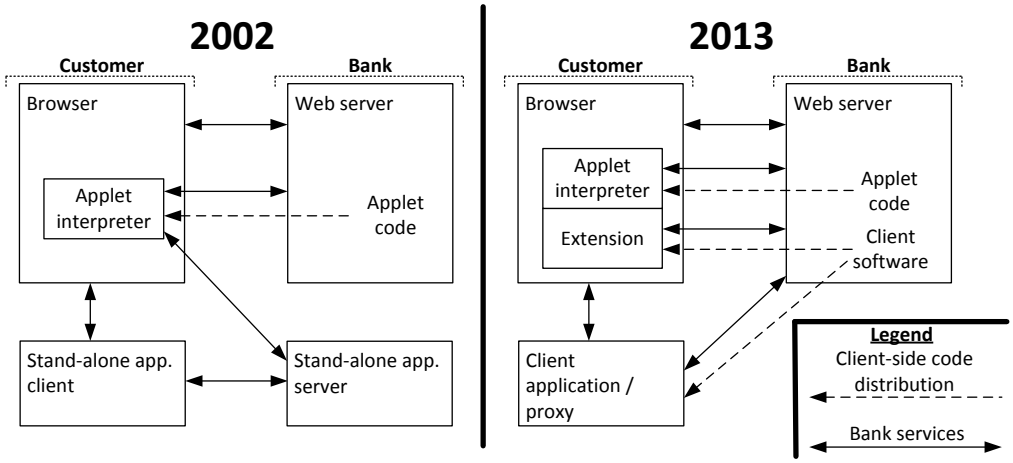

Figure 3: Internet banking structure with PCs.

The most common way for a customer to interact with the bank in 2013 is with the bank's site using a browser on a local machine. 68 banks in our survey do not require additional software. 9 banks require the installation of a proprietary browser extension distributed by the bank and 3 banks require the installation of Oracle's Java Runtime Environment to run applets on the customer's computer. From the 12 banks that distribute client-side code (proprietary browser plugins and Java applets), 10 use client-side code for password or PIN-based authentication while 8 banks (also) use client-side code for certificate-based authentication.

Banks that offer browser-independent client-side software packages today are rare and do not seem to promote it as much as browser-based interaction for online banking. In our survey, only two banks were found that provide client-side software. The software package of one bank is aimed at consumers and provides several functions in addition to those that the web interface also offers. The additional functions include offline examination of bank account data and the ability to proxy a connection from a web browser to the bank's web interface. The included documentation claims that by providing mutual authentication, encryption and data signing through a proxy, the problem of direct and open information exchange through a browser is solved. The software from the second bank bank aims at small to medium businesses with large sets of transactions.

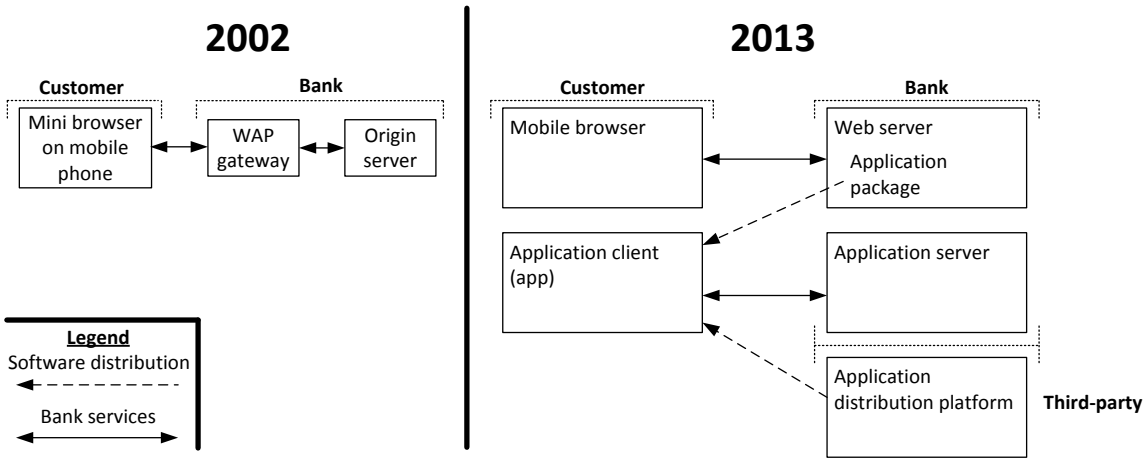

Figure 4: Internet banking structure when mobile computers are used today.

2.3.2. Mobile devices

Figure 4 shows the basic architecture of mobile banking as noted in 2002 by Claessens et al. and in 2013 as noted by us. In our survey, from the total of 81 examined banks there are 67 that provide mobile banking-based services.

In 2013, client applications (or 'apps') are used to communicate with the servers of the bank for balance checking, viewing the transaction history and conducting transactions. In addition to the user and the bank, a third party can be involved in the distribution of the bank's client application to the user's device. The application distribution platform provided by the mobile operating system developer (the third-party) and associated with the used mobile operating system allows the bank to provide the application for free to anyone who wants to install it. Since the application distribution platform is the main method used for installing software by the device's user, it also provides a user friendly and familiar environment when installing the bank application. Examples include Google Play [2] for Google's Android and the App Store [3] for Apple's iOS. 54 of the 67 banks that provide mobile banking-based services offer this through client applications. Of these 54 banks, 50 distribute applications through a third-party channel.

Users can also use a mobile web browser on MDs for online banking. The site normally used by PCs can be visited and used in the same way. The only condition for this is that a browser extension is not required, since browser extensions developed for PCs do not work on MDs. 22 of the 67 banks that

provide mobile services offer HTTP-based sites specifically aimed at MDs. These sites are made for devices with smaller screen sizes and limited input capabilities.

In our survey, access through actual WAP is still provided by 4 banks. WAP 1.0 is a protocol suite used to transfer information between a mobile phone and a gateway in a format that required limited bandwidth. The gateway connects to services worldwide as requested by the mobile phone and translates traffic between the phone (using the WAP protocol suite) and the requested services (in HTTP). A compact markup language known as Wireless Markup Language (WML) was used in addition to WAP-specific protocols for the communication between phone and gateway. WAP 2.0 opted for the XHTML Mobile Profile to replace WML and made the gateway optional (serving as a proxy instead of a translator). Both versions of the protocol suite can be considered obsolete because MDs now have access to the same web technologies as PCs and bandwidth is less of a concern. Due to WAP's legacy as the first well-known standardized way to make mobile devices communicate with the World Wide Web, the term 'WAP' is sometimes used as a name for a normal (HTTP) mobile site. Because of this, some banks incorrectly use the term 'WAP' to refer to HTTP-based sites aimed at MDs. We corrected this in our collected data.

2.3.3. Evolution of PCs

The processing power of PCs increased between 2002 and 2013 and therefore the PC still represents a very powerful class of device. However, the openness of the platform is a rarely used aspect. There are some banks in our survey that still actively develop client-side software in the form of browser plugins or (very rarely) stand-alone client-side software. All other banks only depend on the availability of a browser.

We note that while PCs have become more powerful, that online banking took little advantage of this. Standard web technologies have been used to replace stand-alone programs on the client-side and therefore it is easier for banks to implement security updates, since they only have to be applied server-side. Client-side security updates are left to the customer, and thereby indirectly to maintainers of the operating system and the used browser. This is quite user friendly to customers of online banking since the only software requirement for most banks is a browser, which removes the need to update specific client-side bank software. However, the use of standard technologies also makes it easier for attackers to adjust their strategies to work with dif-

ferent banks. The same standard web technologies are used by most banks, allowing the modification of one attack to work against multiple banks without much effort.

2.3.4. Evolution of MDs

In sharp contrast to PCs is the evolution of MDs between 2002 and 2013. While processing power has increased tremendously in a similar way to PCs, it is the increase in functionality and development options that makes the most noticeable difference between the evolution of both types of devices. All modern MDs have a browser to access most normal banking sites in a similar way to browsers on PCs. These browsers can also access online mobile banking sites specifically made for MDs. Mobile banking sites are adjusted to the often limited input options and screen size of MDs and therefore allow a better user experience on these devices compared to non-mobile sites aimed at PCs.

The openness of the platforms and available development tools allow banks to design applications that have a better integration with both hard- and software, thereby gaining access to additional functionality offered by MDs. Modern MD operating systems apply a technique known as sandboxing, which logically separates the storage and memory space an application uses from other applications [4, 5]. While this separation is only on a software-level (everything still runs on the same hardware), it gives applications more protection from the behavior of other applications compared to PCs.

Updating and maintaining applications on most mobile operating systems is done through a single distribution channel. Already mentioned examples include the Play store for Android and the App Store for iOS. If an update is available, an already existing mobile application can detect this and refuse to work unless the update is installed. This allows banks to keep installations of their mobile applications up-to-date on the MDs of their customers. Bug fixes and security updates can therefore be distributed rapidly through the official distribution channels.

From a security perspective, MDs have evolved to platforms with integrated security controls. Due to their intrinsic security features such as sandboxing and automatic updates, mobile banking applications run in an environment that is intrinsically safer compared to the application environment that is offered by PCs. Because there is a larger fragmentation in client-side code between mobile banks, it is harder to attack different banks

using their offered mobile applications. While this does not have to be a conscious decision, it reduces the impact of attacks and therefore improves security.

2.3.5. Security implications of modern MDs - mobility and authentication

Using the knowledge authentication factor commonly requires a user to enter information in a device. This is usually in the form of a fixed password. Due to both their small size and limited input options (often a touch screen or a keypad), MDs often apply techniques that reduce the number of characters a user needs to enter to input common words or phrases. These techniques are not used for password fields because their ability to learn new words and phrases from a user creates the risk that the user's password is added to a technique's dictionary. Passwords have to be typed manually by the user, complying with password restrictions (such as having a specific length and the use of lower and upper characters, numbers and special characters) that might be cumbersome to enter on MDs.

The relation between perceived input (what the user thinks he/she enters) and actual input (what is received by the device) is not as strong with MDs as it is with PCs due to the cumbersome input methods. For password fields, mobile operating systems show each character as entered before hiding it (either after a timeout or after another character is entered). Because passwords have to be entered completely, it is easier for someone in close proximity to the user to read the password from the screen. To increase usability, passwords restrictions are often more relaxed for MDs compared to PCs. Instead of a password, a PIN is used in some cases to further simplify character entry. Both the observation of characters on the screen of the device when they are entered in the device and the shortening of passwords are examples of a reduction in security to gain usability with MDs.

Another security implication is presented by the physical location of MDs. Where PCs are often used in a more private or controlled environment (at home or at the office) due to their physical size, MDs are often also used in public places. Not only makes this fixed passwords more prone to simple 'passive' observation (also known as shoulder surfing). It also allows an attacker to pickpocket the MD after the user was observed entering a fixed password, which can be required to use mobile banking with the bank account of the device's owner. Also, more mobility presents a higher probability to accidentally lose a device.

As noted, MDs have their limitations in terms of security. But there

is also an area in which MDs excel with online banking compared to PCs. With mobile banking applications, a customer can register a mobile device to a bank account. Registration is done once and requires either a visit to the bank or the same authentication that is normally used for banking with PCs. After registration, a user friendlier method (often a PIN) can be used by the customer to access the bank account on the device it was registered to. The binding of a customer's bank account with a specific MD effectively makes online banking on one's own smartphone or tablet use multi-factor authentication, because the specific MD ('something you have') and the PIN ('something you know') both need to be available. This increases user friendliness, since a customer is not required to carry an additional authentication device for online banking.

Still, the mobility of an MD allows an attacker to simply steal and use an MD after looking over a customer's shoulder for the required information to get access to the customer's online bank account. Some banks mitigate this by requiring an extra form of authentication when money transfers are made to new destination account numbers. Unfortunately, the extra authentication step used by some banks is a text message with a one time password (OTP) that can be send to the same smartphone on which the mobile bank application is running, thereby making the extra authentication step nullify any added security advantages if an attacker already has access to the device.

Mobility definitely brings advantages in functionality in all observed cases. However, possible advantages in security are not always exploited by banks and disadvantages are not always properly mitigated. We note that developments in mobile online banking are first and foremost driven by usability and not by security.

2.3.6. Security implications of modern MDs - software and hardware security

From a security perspective, MDs have evolved to platforms with integrated security controls. Due to their intrinsic security features such as sandboxing and managed automatic updates, mobile banking applications run in an environment that is intrinsically safer compared to the application environment that is offered by PCs. Because there is a larger fragmentation in client-side code between mobile banks, it is harder to attack different banks using their offered mobile applications. We did not do a thorough examination of all mobile applications, but it is trivial to see that the underlying structure between applications differs greatly. Diversity makes it difficult to attack multiple online banks through their mobile applications

using a single strategy. This is the opposite of online banking using PCs, where the same standard web technologies are used by most banks, allowing the modification of one attack to work against multiple banks. While this diversity does not have to be a conscious decision, it reduces the impact of attacks and therefore improves security.

With PCs, most banks do not depend on the used operating system (unless they require the use of browser plugins, which often do depend on the used operating system). There are not many technical demands specific to online banking, as long as the operating system and the browser receive security updates. Most modern PC operating systems and browsers automatically apply updates or urge the user to update.

The situation is different for MDs. Banks that support mobile banking depend on the security of mobile operating systems. Newer versions of a mobile operating system contain new features and security updates not only for the operating system itself, but also for the included browser. It is up to the developer of the operating system to provide updates, up to the manufacturer (and in some cases also the mobile service provider) to distribute these updates and up to the end-user to finally install them. Due to the combination of these dependencies, a considerable number of Android devices in use do not have the latest version of the operating system installed and therefore miss important security updates. Mansfield-Devine [6] notes that in 2012, 98.2% of the used Android devices do not run the latest version of the operating system. According to the release dates of new Android versions, this percentage of devices were at the time using a mobile operating system that was at least six months old. An overview of which Android versions are in use one year later is given in Figure 5. The data is based on devices that use Google's Play Store.[9]

98.5% of all Android devices in use on 2 October, 2013 did not run the latest version of the mobile operating system. Android 4.3 was only released a few months earlier.[10] When we consider that manufacturers might not have had time to distribute this new update and therefore state that devices with Android 4.2 can be seen as up-to-date, then of all devices there is still 87.9% that run a version of Android that is lower than 4.2. The first version

[9]Android versions in use as registered by Google Play Store: `http://developer.android.com/about/dashboards/index.html`

[10]The Verge on the deployment of Android 4.3 on 2013-07-24: `http://www.theverge.com/2013/7/24/4550234/android-4-3-announcement`

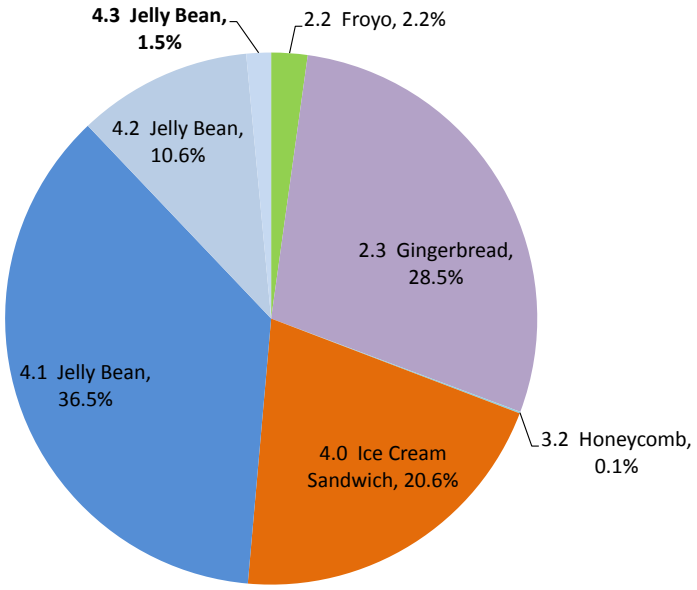

Figure 5: Android versions in use, as of 2 October, 2013.

of Android 4.2 was released 11 months earlier.[11] In other words, 87.9% of all Android devices are using an operating system that is at least 11 months old.

The numbers look better for iOS devices. According to Apple, only 7% of all customers that use iOS devices that connect to Apple's App Store were using an older operating system in June 2013 [7]. The large difference in the use of older versions between Android and iOS can be explained in that the update process of iOS is less complex. Android updates rely on the operating system developer (Google), the device manufacturer (Samsung, HTC, Huawei, etc.), possibly the mobile network operator if it concerns a 'branded' device and the end user. iOS, in contrast, only depends on Apple (as an operating system developer and device manufacturer that does not allow interference from a mobile network operator) and the end user. Apple is also quite aggressive in pushing updates to its users and thereby keeping the amount of fragmentation on iOS devices as little as possible.[12] Due to the

[11] The Android Soul - Android 4.2 update scheduled for release on 2012-11-13: `http://www.theandroidsoul.com/android-4-2-update-for-nexus-7-confirmed-for-nov-13-galaxy-nexus-is-scheduled-for-later/`

[12] Rowinski (2013) - Why Apple's iOS Could Benefit From

combination of these dependencies, a considerable number of Android devices in use do not have the latest version of the operating system installed and therefore miss important security updates [6]. Manufacturers that maintain stricter version control of installed operating systems seem to do it better. According to Apple, only a small number of devices that have iOS do not run the latest version [7].

While mobile operating systems have a number of intrinsic security features, a lack of updates can form a security issue. Banks need to keep their applications compatible with older versions of mobile operating systems to satisfy owners of MDs that do not receive updates. This opens mobile banking applications to vulnerabilities when they are used on an older operating system. The same is true for mobile browsers included with mobile operating systems, which do not get updated when the mobile operating system itself is not updated.

As for hardware, MDs have not reached their full potential for security improvements. A Trusted Execution Environment allows applications to be executed in a way that protects the confidentiality and integrity of the application itself and the resources it uses from other applications that run on the same platform. Van Rijswijk-Deij and Poll explains in [8] the currently applied TEE technologies in MDs: Intel's Identity Protection Technology (IPT) and ARM's TrustZone. They note significant drawbacks in both technologies and conclude that current TEE implementations do not offer a direct replacement for existing two-factor authentication methods. Both technologies are proprietary and it is difficult to find detailed public documentation. These drawbacks, combined with a fragmented market (not every MD offers a TEE environment and offered environments can differ greatly from each other), hamper the acceptance of TEEs in their current state.

2.4. Communication security

In 2002, the standard solutions for communication security with online banking were Secure Sockets Layer/Transport Layer Security (SSL/TLS) for PCs and Wireless Transport Layer Security (WTLS) for MDs. Claessens et al. describe in [1] the various versions of SSL/TLS up to SSL 3.0 [9] and TLS 1.0 [10]. Since then, several weaknesses in SSL/TLS implementations

A Little Fragmentation: `http://readwrite.com/2013/10/18/ios-7-downgrade-bugs-updates-battery-drain-fragmentation`

have been discovered and new versions of TLS have been developed (TLS 1.1 [11] and TLS 1.2 [12]). An update was also released for all TLS versions that breaks backwards compatibility of TLS with SSL 2.0 [13]. WTLS has not seen a newer version since 2001 [14], which can be explained by that WAP usage has strongly declined in favor of standard web technologies on MDs such as SSL/TLS.

We examined the use of SSL/TLS by the banks in our survey. Due to the large number of banks, we narrowed down our analysis to the use of SSL/TLS to secure communication between non-mobile banking sites and web browsers (HTTPS). The use of SSL/TLS between mobile applications and bank servers was not examined. Banks that still apply WTLS were also not examined.

The authors of this report do not have accounts at most banks, which is why SSL/TLS-usage was by examined using the login pages. 78 of the 80 banks surveyed for online banking rely on SSL/TLS for communication security, at least for the authentication process. The 2 exceptions in our survey send all information unencrypted, including authentication credentials.

Whether the 78 banks implement SSL/TLS correctly is something that can be evaluated. We tested for the supported SSL/TLS versions and the presence of several weaknesses and vulnerabilities in used certificates, cryptographic algorithms and SSL/TLS configurations. Almost all SSL/TLS bank sites were analyzed using Qualys SSL Labs SSL Server Test [15]. One bank did not use the default TCP port for HTTPS communication (443). Non-default ports are not supported by Qualys' analyzer, which is why we used COMODO SSL Analyzer [16] instead.

The examined weaknesses and vulnerabilities are as follows:

- Invalid certificates. When a site presents a certificate to a browser, several aspects of the certificate are tested. The certificate is deemed invalid if one of these tests fails and with that, the site is deemed untrusted. Modern browsers give a warning when an invalid certificate is presented, but allow the user to ignore this. The tested aspects:
    1. Site name. A server certificate is given a DNS name (such as `example.com`). Optionally, the wildcard character `*` can be used to indicate that a certificate is valid for a domain and all its subdomains (such as `*.example.com`). If the site that presents the certificate was not requested using the DNS name stated on the certificate, the certificate is invalid.

2. Expiration date. Every certificate has a date after which the certificate loses its validity.
3. Trust chain. The trust in a certificate is based on the chain of certificate authorities (CAs) that were (indirectly) involved in signing it. A root CA identifies itself with a certificate that is self-signed. An intermediate CA identifies itself with a certificate that is either signed by a root CA or by another intermediate CA. When a browser tests the trust chain of an offered certificate, it first checks whether the certificate is signed by a trusted root CA. A root CA is trusted if its certificate is included in the trusted root certificate store used by the browser. If the offered certificate was not signed by a trusted root CA, the browser checks whether the certificate of the signing intermediate CA is signed by a trusted root CA. This process repeats for every higher level CA of which the certificate can be retrieved until the chain ends (there are no more certificates to check), in which case the offered certificate is untrusted, or until an intermediate CA certificate is found of which its certificate is signed by a trusted root CA.

- Weak hash functions. A certificate consists of several parts, including information about the party it identifies (such as the party's public key), an identifier of the issuer, a hash value (or message digest) to uniquely identify the certificate and a signature by the issuing party. The signature depends both on the hash value and on the issuer's private key, and therefore respectively on the used hash function and asymmetric key algorithm.

  It is called a collision when two distinct pieces of data compute to the same hash value. A hash function used for signature algorithms is weak if the distinct data required for a collision is easy to find. The required amount of time to find a collision depends on the hash function's collision resistance. The issuer's private key is used to sign the hast value. If a collision is found by changing the data without changing the resulting hash value, it is possible to create a forged certificate with the same (valid) signature from the issuer.

  Several hash functions that were used for digital certificates are not collision resistant. An example is MD5 [17], which should not be used for signature generation since it is trivial to find collisions and there-

fore cryptographically broken [18, 19, 20]. There were already signals that MD5 was weak against collision attacks in 1996. At that time, it was recommended to switch to SHA-1 [13] or RIPEMD-160 [14] [21]. While a successful actual (in the wild, not in a test environment) collision attack against SHA-1 has not yet been observed, it is expected that such an attack will be seen in the future due to several weaknesses in the algorithm [22]. The use of SHA-1 for digital signature generation (and with that, its use in signing digital certificates) is not recommended after 2013 according to NIST, which recommends the use of SHA-2 (SHA-224, SHA-256, SHA-384 or SHA-512) instead [23]. Some alternatives which have not (yet) seen a widespread role in digital signatures include the already mentioned RIPEMD-160 and the relative young Keccak, which is selected by NIST to be the new SHA-3 standard.[15]

For our survey, we will count the use of MD5 as a vulnerability. SHA-1 usage will not be counted as a vulnerability because in 2013 it is still a recommended hash function.

- Weak and insecure ciphers. SSL/TLS supports a number of ciphers with different key sizes to support the confidentiality and integrity of an established session. If a key size is weak or a cipher is insecure, brute-force attempts to retrieve a key based on captured data requires less time and resources compared to when a strong combination of key size and cipher is used.

  The applied SSL Server Test from Qualys designates all cipher suites that are less than 128 bits as 'weak' when an SSL site is tested. If the assumption is made that the data has to stay confidential and its integrity safeguarded against eavesdroppers for the period '2031 and Beyond', the minimum of 128 bits conforms with recommendations by NIST [24].

  A cipher suite can also be noted as 'insecure' by the SSL Server Test

[13] FIPS 180–4, describing SHA-1, SHA-224, SHA-256, SHA-384 and SHA-512: `http://csrc.nist.gov/publications/fips/fips180-4/fips-180-4.pdf`

[14] The hash function RIPEMD-160: `http://homes.esat.kuleuven.be/~bosselae/ripemd160.html`

[15] The Keccak sponge function family: `http://keccak.noekeon.org/`

from Qualys. This indicates that it does not provide authentication (with anonymous Diffie-Hellman cipher suites) or that it does not provide encryption (with NULL cipher suites) [25].

- SSL 2.0. The first public version of SSL has a number of flaws that were already acknowledged in 2002 by Claessens et al. in [1]. These are the use of the same cryptographic keys for message authentication and for encryption (which makes the security of Message Authentication Codes (MACs) unnecessary weak when encryption key size is limited due to export restrictions), the sole dependence on MD5 as a vulnerable hash function to construct MACs, the lack of handshake protection and the possibility to truncate a connection due to relying on the closure of the TCP connection. Supporting SSL 2.0 is not recommended, since all modern browsers support SSL 3.0.

  While SSL 2.0 might still be supported in browsers, it is by default disabled and not easily enabled by end-users. Examples of popular browsers that do support SSL 3.0 and have SSL 2.0 disabled include Microsoft Internet Explorer since version 7.0 Beta 2 [16], Mozilla Firefox since version 1.8.1 beta 1[17] and Opera since version 9.5 [18].

- Vulnerability to Browser Exploit Against SSL/TLS (BEAST) attacks. Within the SSL/TLS protocol suite (up to versions 3.0/1.0, respectively), one method to encrypt data is with block ciphers (cipher-block chaining, CBC) using symmetric-key encryption. The SSL/TLS standard mandates that chained initialization vectors (IVs) are used with CBC mode encryption. With chained initialization vectors, the last block of the previous ciphertext is used as an IV for the next message. This presents a vulnerability that can be exploited using a blockwise chosen-boundary attack (BCBA) [26]. A BCBA applied on a HTTPS session is known as a BEAST attack [27].

  The best defense against BEAST is the use of TLS 1.1 or 1.2, which

[16] Microsoft disables SSL 2.0 by default in Internet Explorer: `http://blogs.msdn.com/ie/archive/2005/10/22/483795.aspx`

[17] Mozilla disables SSL 2.0 by default: `https://bugzilla.mozilla.org/show_bug.cgi?id=236933`

[18] Opera disables SSL 2.0 by default: `http://www.opera.com/docs/changelogs/windows/950/`

fixes the exploit by using explicit (pseudo-random, instead of basing it on information of the previous block) IVs.[19] This requires support from both web servers and browsers. A proposed alternative when TLS 1.0 is required is prioritizing RC4 cipher suites, thereby choosing stream ciphers (which are not vulnerable to BEAST) over block ciphers. Unfortunately, RC4 is known for its own vulnerabilities.[20]

Another alternative is mitigating the attack by implementing 1/n-1 record splitting as a workaround in browsers. With this, a CBC encrypted record is broken in a part with a singly byte and a part with the rest of the record. This effectively randomizes the IV and protects against the attack.[21] This fix has been implemented in most browsers, mitigating the currently known attack vector client-side.[22]

- Vulnerability to Compression Ratio Info-leak Made Easy (CRIME) attacks. If an attacker can observe network traffic and manipulate a victim's browser to submit requests to a target site, it is possible to retrieve data from the TLS stream when DEFLATE compression is used. An attacker can steal session cookies with CRIME, which makes it possible to hijack a session [28].

  While this attack is easier to execute compared to BEAST, it is also easier to defend against by disabling TLS compression. This can be done server- or client-side. The vulnerability only exists when both server and client support and use TLS compression when a session is established.

The SSL Server Test by Qualys only uses the responses from servers to normal SSL/TLS requests. Other vulnerabilities exist, but to test servers for them requires more extensive and intrusive requests.

An example is the Lucky 13 attack [29]. Lucky 13 attempts to retrieve plain text that is crucial to secure sessions (such as cookies or passwords) from

[19] RFC 4346, TLS 1.1, Explicit IVs: `http://tools.ietf.org/html/rfc4346#appendix-F.3`

[20] On the Security of RC4 in TLS and WPA: `http://www.isg.rhul.ac.uk/tls/`

[21] Adam Langley - BEAST followup: `https://www.imperialviolet.org/2012/01/15/beastfollowup.html`

[22] Qualys Security Labs - is BEAST Still a Threat?: `https://community.qualys.com/blogs/securitylabs/2013/09/10/is-beast-still-a-threat`

cipher text by measuring the timing differences between different SSL/TLS responses. The difference in timing comes from the plain text and the steps required to encrypt it, which gives (when applying iteration of the attack and statistical analysis) enough information to retrieve the plain text data.

Vulnerabilities against attacks on the level of Lucky 13 require testing that involves more than normal(ly applied) SSL/TLS requests, which can be considered intrusive by the surveyed banks. We refrained from testing these vulnerabilities in our survey because of this.

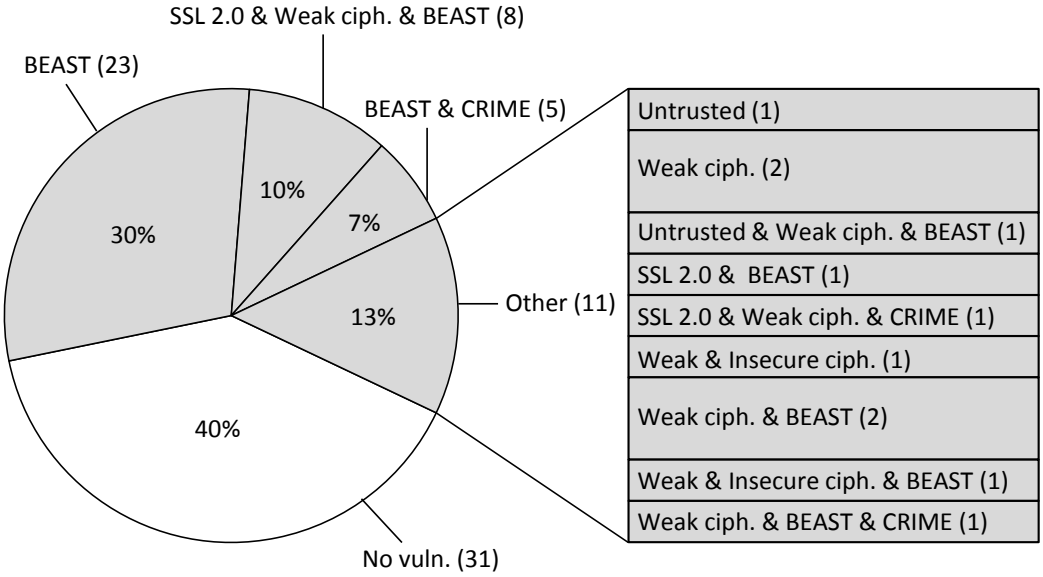

Figure 6: An overview of the encountered SSL/TLS vulnerabilities.

Figure 6 shows an overview of the vulnerabilities and how they are distributed among the surveyed banks. Only 40% of the banks have correctly configured web servers with correct certificates. 46% of the servers is vulnerable to BEAST attacks while 7% is vulnerable to both BEAST and CRIME. 13% of the surveyed servers still support SSL 2.0. Some servers are not vulnerable to SSL 2.0 attacks because they do not have supported cipher suites for this version of the protocol. Since these servers are not vulnerable, they were excluded from our count of servers that are vulnerable to SSL 2.0 attacks.

An overview of supported SSL/TLS versions by the examined banks can be found in Figure 7. Servers that support SSL 2.0 but are not vulnerable to

relevant attacks have the symbol * in their SSL version description. Servers that are vulnerable are represented in black.

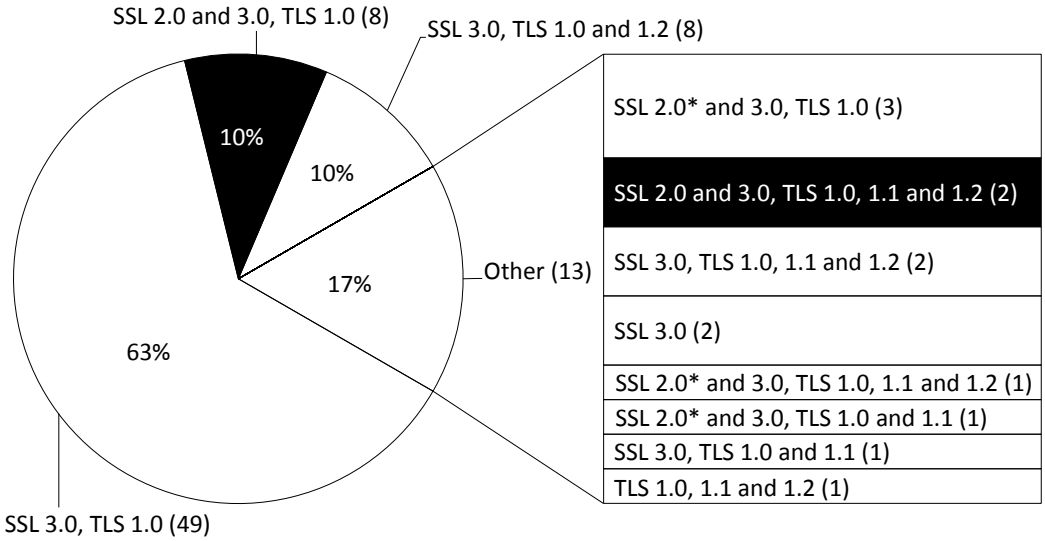

Figure 7: Supported SSL/TLS versions among the surveyed banks.

Not shown in Figure 6 is the use of weak hash functions, since this was not found at a single bank. 77 banks use SHA-1 for generating a hash that is signed with RSA to form the signature. The exception was a bank that used SHA-512 with RSA. This is most likely a mistake since that particular bank used a self-signed certificate that refers to a test environment instead of their public site.

2.4.1. Extended Validation

Another aspect that we tested was the use of Extended Validation (EV) certificates. From a technical perspective, EV is an extra attribute added to a certificate when issued by a certificate authority. Users can be notified by their browsers while visiting a SSL/TLS site that an EV attribute is present in the certificate offered by the server. If and how users are notified depends on which browser (version) and what type of device (PC or MD) is used. From a functional perspective, it is expected that more administrative procedures have to be followed for a CA to issue EV certificates in order to gain a higher level of trust in the identity of the certificate's owner. The expected procedures are noted in guidelines as published by the CA/Browser forum in [30].

EV depends on the capabilities and willingness of users to recognize the difference between basic certificates and EV certificates. Whether EV provides any benefit is disputed. Without training or guidance, a considerable number of users do not notice the differences between offered basic and EV certificates in web browsers [31, 32, 33].

In our survey, the SSL/TLS configuration and certificates from 78 bank sites are tested. 76 of the sites offer certificates that are trusted. From these 76 sites there are 50 that offer an EV certificate.

2.5. Authentication methods

A customer must perform authentication to prove their identity to a bank before a session is initiated in which bank account(s) can be managed. This is referred to as entity authentication. Furthermore, it is possible that an extra authentication step is required to authorize the transfer of money. This is called transaction authentication. Entity authentication is mandatory while transaction authentication is optional to implement [1].

Several factors can be used in either form of authentication. These are knowledge (something the customer knows), possession (something the customer physically has) and existence (something the customer physically is). The terms two- or multi-factor authentication are used when at least two different factors need to be fulfilled to establish an authenticated session. With online banking only knowledge and possession are represented. Existence can be measured using biometrics, but was not applied at any bank in our research. Possible reasons for this are that biometrics cannot be used by everybody and restrictions in social acceptability, thereby reducing user satisfaction below the acceptable threshold [34]. Another possible explanation is that it is too expensive since specialized hardware has to be distributed to each customer.

2.5.1. Knowledge

The customer's knowledge (i.e. something the customer remembers) is used as a single or additional factor by most banks. Observed forms are passwords and PINs. Passwords use alphanumeric and sometimes special characters. PINs are quite similar, but only allow numerical characters with stricter demands on length. If only a password or PIN is required, it is unlikely to be used for transaction authentication since it only reconfirms the authentication credentials used to establish the session with the bank. In this case, it does not add much to security since it is prone to a replay

	Possession factor			
Knowledge factor	Unavailable	Optional	Mandatory	Unknown
Password	16	8	25	0
PIN	3	1	14	0
Password and PIN	1	0	5	0
Password or PIN	0	1	3	0
None	0	0	3	0
Unknown	0	0	0	1

Table 1: Which type of knowledge factor is applied by banks and whether these are combined with a type of possession factor (PCs). It could not be determined which authentication method(s) were used at a single bank.

	Possession factor			
Knowledge factor	Unavailable	Optional	Mandatory	Unknown
Password	10	0	3	0
PIN	4	1	0	0
Password and PIN	1	0	0	0
Unknown	0	0	0	3

Table 2: Banks that apply the knowledge factor and whether these are combined with the possession factor (MDs, mobile sites). The authentication methods of 3 banks were unknown because they could not be examined through documentation or trial.

	Possession factor			
Knowledge factor	Unavailable	Optional	Mandatory	Unknown
Password	13	0	9	2
PIN	6	0	16	3
Password and PIN	1	0	0	0
Unknown	0	0	0	4

Table 3: Banks that apply the knowledge factor and whether these are combined with the possession factor (MDs, apps). Authentication methods could not be examined at 4 banks due to a lack of documentation or installation opportunity. The knowledge factor could be identified at 5 banks, but it was unknown if an additional factor would be required after login.

Type	Offline	Online	Offline or online	N/A
Physical	8	-	-	-
Stand-alone token	14	0	0	-
Smart card + Reader	3	0	2	-
SMS	-	16	-	-
Physical or SMS	-	-	2	-
Physical and SMS	-	-	1	-
Physical or stand-alone token	1	-	0	-
Physical, smart card + reader or SMS	0	0	1	-
Smart card + reader or SMS	-	1	0	-
None (do not use OTP)	-	-	-	31

Table 4: How banks use OTPs to authenticate customers who use PCs.

attack. If an attacker knows the password to establish a session, asking the same password to authorize transactions does not add anything to security.

An overview of which knowledge-based methods the banks use and whether they combine them with possession-based methods is given in Table 1 for PCs. Table 2 and Table 3 show the same values for mobile sites and mobile apps, respectively. Note that only authentication methods are counted that provide the maximum available functionality (e.g. the ability to create transactions to new recipients) to the customer.

2.5.2. One-time passwords and challenge-response authentication

A one-time password (OTP) differs from a fixed password in that the former can only be used once. OTP authentication is used for entity authentication and transaction authentication, but rarely for both at the same bank. It is common for banks to use OTPs for transaction authentication when a password or PIN is used for entity authentication.

Customers can acquire valid OTPs in several ways. OTPs can be distributed physically, or they can be electronically acquired on a need-to-use base instead. Physical OTPs are distributed as a list on paper or as a character grid on a plastic card, and therefore always offline. Electronic OTPs are distributed through devices. The number of banks in our research that apply challenge-response authentication and the types of used devices are given in Table 4.

Challenge-response authentication requires that the to-be authenticated party gives an expected reply (response) based on information given by the

Type	Offline	Online	Offline or online	N/A
Stand-alone token	4	0	0	-
Smart card + terminal	5	0	2	-
None (do not use c-r)	-	-	-	69

Table 5: Banks that use challenge-response to authenticate customers and/or transactions on PCs.

authentication facilitating party (challenge). This type of authentication is often used for transaction authentication. With online banking, a challenge is generated by the bank for the customer to present to the authentication device. In turn, the authentication device generates a response based on the challenge, to be sent back to the bank. The bank has the ability to test whether the given response is valid. If it is, authentication is successful. The number of banks in our research that apply challenge-response authentication and the types of used devices are given in Table 5.

For our article, we use the term 'token' to refer to an electronic device used by a customer for OTP and/or challenge-response based authentication. We consider a 'security token' as a different type of device with the sole purpose to aid public-key cryptography by hosting the private key and any functionality that requires the private key. Most token devices issued by banks that generate an OTP require a PIN before OTPs are given. When challenge-response authentication is used, it is always required to enter a PIN code before a device accepts the input of challenges to compute responses.

Tokens come in offline and online forms. Offline tokens require that the customer enters information from the token in the bank's web interface (and with challenge-response authentication also vice versa).

Online tokens do not require that the customer facilitates the information exchange completely. Communication is electronically facilitated in one or both directions. A second or out-of-band communication channel can be used by the token to communicate with the bank. An example of a semi out-of-band channel (due to that it is used uni-directional) is the use of SMS text messages that are sent to the customer's phone. These contain OTPs to be entered by the customer in the interface of the bank. Several banks offer examples of bi-directional communication with readers for smart cards that are connected to the customer's computer. User input is either completely facilitated by the customer's computer, or several important steps (such as PIN entry and action verification) need to be followed using a screen

and keypad on the reader. Online tokens that involve smart cards can also optionally be secured by a PIN that either has to be entered in the computer itself or on the reader device.

A further distinction can be made between tokens. We observed two types of devices in our survey: stand-alone tokens and smart cards that are used with (mobile) terminals. The former type is only observed for offline tokens, while the latter is also sometimes used for online tokens.

The only purpose of a stand-alone token (as seen in our survey) is to work as an authentication device for online banking. A bank card is used used for conventional electronic payments, such as in physical stores. A stand-alone token can have its own PIN that is independent from the PIN of a bank card from the same customer. This distinction of functionality and authentication credentials is an advantage from a security perspective, because the compromise of one device and its associated functionality does not compromise the other. Even if the PINs would be the same on both devices or one device does not have a PIN (e.g. a one-button token for online banking), the associated functionality with each device is still distinct and therefore only compromised if the corresponding device is compromised.

The advantage of functional and credential separation is lost when a smart card is used with a terminal (a battery-powered device with a display, keypad and smart card slot) instead of a stand-alone token. With the banks in our survey who use smart cards with terminals, the smart card that holds the authentication credentials for online banking is the same bank card that is used for conventional payments. When the card and its PIN are compromised, the attacker can use the card for conventional payments and online payments. Terminals are not personalized and can be used for online banking with any smart card from the bank that issues them, excluding them as a possession factor in authentication.

2.5.3. Certificate-based authentication and platform binding

Certificate-based authentication allows a bank to verify a customer's identity by making the customer prove that they own a digital certificate. In our survey, this type of authentication is used for both entity and transaction authentication.

A distinction on the level of protection of the private key that is owned by the customer can be made, which is stored in either software or hardware and optionally protected by a PIN or password. If the private key is stored in software, no other devices are necessary to facilitate the use of the key. The

opposite is true for hardware, in which the private key is stored in a separate device (a security token or smart card) and actions that directly involve the private key are done by the supporting hardware, outside the software environment of the main platform. This prevents exposure of the private key to the software environment, thereby reducing the risk of retrieving and using the key on other devices.

9 banks in our survey support the use of certificate-based authentication by customers for PCs. Of these, 4 allow the storage of the key in either software or hardware. 3 banks only support software and 2 bank only support hardware. Support for key management on the customer's computer is either provided through proprietary software (8 banks) or through functionality provided by the web browser using SSL/TLS client-side authentication (1 bank). Of the 8 banks, 5 distribute software to support hardware that holds the private key and calculates responses. Based on a limited examination of this software, we conclude that the hardware itself is most likely proprietary and unknown to us.

Of the 67 banks that provide mobile services, 54 provide mobile applications. 18 banks restrict the use of these applications to devices that underwent a registration process initiated by the customer. The mobile application is bound both to the device and to the customer's bank account. This adds the mobile device itself as an authentication factor (possession). We did not examine how this binding is done on a technical level due to that debugging applications is time-consuming and the gained knowledge is limited in the context of this article.

We found one bank in our survey that applied platform binding for the secure mobile site it offered to its customers. After a customer is logged in for the first time on the mobile site using the same authentication method that is used for the normal site (involving OTP authentication using an offline token), a cookie is placed on the mobile device. When revisiting the site, the mobile browser presents the cookie and the customer only has to use a PIN code to log in again. Transactions to new destination account numbers still require challenge-response authentication, but other actions can be done in the logged in session without any further authentication.

2.5.4. Other authentication and verification methods

Some banks in our survey use methods for online banking using a PC that seem quite exceptional and rare compared to the frequency of other examined methods.

File verification
A rarely used method that we found in our survey is the use of a file provided by the bank and stored by the customer's computer. Whenever the customer initiates an action that requires extra security (such as the transfer of money to a third party), the site requests the file. If the file is provided by the customer, the operation is permitted and a new file is provided for the next action. The file can be stored on the customer's computer itself or on removable media.

Call verification
Another rare method that we examined was the use of mandatory automated phone calls to verify certain customer actions. After hearing which action is prepared, the customer can confirm the action by entering the four digits shown in the online banking session using the phone's keypad. Call verification is required when money is transferred to a new recipient for the first time, when creating a new standing order (the automated transfer of money to a specific recipient at regular intervals) or when initiating an online credit card payment.

Software-based token on MD
In our survey, two banks provide an alternative authentication method to their default method (a combination of a password and an OTP received by SMS). Instead of receiving an OTP by SMS it is possible to retrieve an OTP from an application on an MD. When an authentication application is installed on the MD, a serial number and activation number must be entered as the first step to register the application to the user's account. These numbers are provided through the online banking environment. In the second step, the MD generates a registration code which must be provided in the online banking environment to complete registration and activate the token. One bank uses this as a fully functional alternative while the other limits functionality to low-risk actions once authentication is successful.

2.5.5. Comparing 2002 with 2013

Table 6 gives an overview of observed basic authentication methods used for online banking in 2002 by Claessens et al. in [1] and in 2013 by us. Check marks indicate that a method was observed and percentages (if the required information is available to produce them) indicate the relative number of

Method	2002	2013
Password/PIN-only	✓(66.7%)	✓(23.8% + 13.8%)
OTP (physical)	✓(6.7%)	✓(16.3%)
OTP (token)	✓	✓(13.8%)
OTP (SMS)		✓(26.3%)
Challenge-response (token)	✓	✓(25%)
Certificate-based	✓(14.3%)	✓(3.8% + 2.5% + 5%)

Table 6: Overview of similarities and differences in basic authentication methods for online banking using a PC as observed in 2002 and 2013. Percentages represent the relative amount of observed banks that apply a method in a specific year. Observations can be compared between years (columns) but not between methods (rows) due to that some banks in our survey are counted multiple times when they offer multiple methods (which is why the percentages can exceed 100% when summed).

banks that apply the methods in the survey for a specific year. Note that all methods except 'Password/PIN-only' are not exclusive. For example, a bank that applies OTP authentication in any way can still require an additional password or PIN from the user. A bank that applies OTP authentication using a token for entity authentication can also apply challenge-response authentication for transaction authentication.

For Password/PIN-only, two percentages are given for our work. These stand respectively for the percentage of banks that offer Password/PIN-only based authentication without any alternative and for banks that provide do provide an alternative. For certificate-based authentication, our percentages relate respectively to the relative number of banks that offer this type of authentication with the private key stored in software only, in hardware only and for the banks which give this as a choice to the user (by supporting both hard- and software storage of the private key).

We excluded the exceptional methods found in our survey to keep the table easy to read. While the often used basic authentication mechanisms show little difference, the ratios in which they are implemented differ between 2002 and 2013. Claessens et al. conclude that in their survey passwords and PINs as a single factor were most widely used to authenticate users. Today, most banks require multi-factor authentication or provide the user with an option. A large number of online banks has since 2002 migrated to methods which are safer but also have been available for quite some time. The only method that is popular now that was not discussed by Claessens et al. in

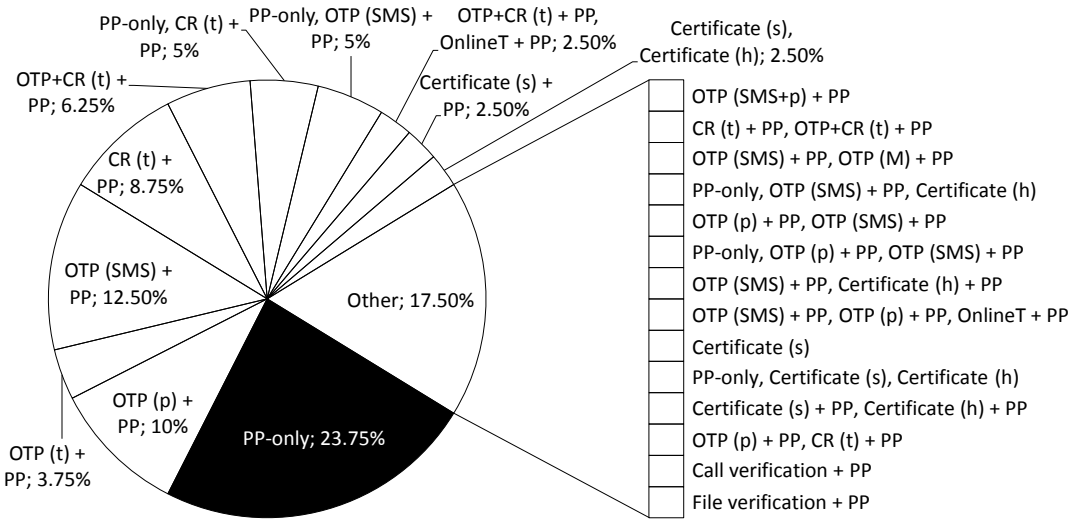

Figure 8: An overview of all combinations of authentication methods observed in 2013. Multiple combinations are separated by commas.

2002 is the use of text messages to send OTPs to customers.

A complete overview of all observed authentication methods and their combinations in 2013 is given in Figure 8. The used abbreviations are described in Table 7.

2.5.6. Trends in authentication methods

We conclude that most observed basic authentication mechanisms for online banking using a PC in 2013 hardly differ from those known in 2002. Mobile banking was quite uncommon at the time, and therefore we base our conclusions on PC-based banking only.

From a security view, the best practice observed in our survey is the use of automated call verification to explicitly ask the customer whether prepared transactions should be authorized to new recipients. Authorization is given by the customer using a four digit PIN code shown on the customer's computer screen. With this, wary customers are given the ability to recognize whether illegitimate transactions to unknown account numbers have been prepared without their knowledge. While the used medium (traditional phone or GSM) is relatively insecure from a communication security point of view, it requires considerable resources to intercept the audio communication between individual customers and their banks. It is also a defense against social engineering, because the customer has to be contacted by the

Method	Medium	Description
PP		Password and/or PIN
OTP	(SMS)	One-time password, sent by text message
	(m)	", computed by mobile phone
	(t)	", computed by token
	(p)	", distributed physically (plastic/paper)
	(SMS+p)	", combined mediums (both mandatory)
CR	(t)	Challenge-response by token
OTP+CR	(t)	Both methods by token
Certificate	(s)	Certificate-based, private key stored in software
	(h)	", private key stored in hardware
OnlineT		Online token (underlying methods unknown)
Call verification		Verify authentication by call to customer
File verification		Use a file as a shared secret

Table 7: Legend for Figure 8.

bank whenever a transaction to a new recipient has been received. To change the phone number on record at the bank, the customer has to contact the bank for identity proving. If the bank does not have the phone number of the customer on record, the customer will have to visit the bank to have it registered for this service.

The observed best practice is only applied by a single bank All other observed (combinations of) methods do not seem to apply explicit out-of-band verification for new money transfer recipients. The second best practice is the use of stand-alone tokens for OTP and challenge-response authentication. This creates a separation of functions between bank authentication devices since the bank cards given to the customers are only used for conventional electronic bank transactions and the given tokens are only used for online banking. Retrieving one (and its corresponding PIN) does not compromise the functionality provided by the other.

Our observations also allow us to make conclusions about the worst practices. While authentication using only a password or PIN has been reduced significantly between 2002 and 2013, it still is offered by a large number of banks optionally or as the only authentication option for online banking. This remains the weakest observed form of authentication, since it allows an attacker to obtain authentication credentials using passive attacks (such as

key logging or wiretapping of unsecured connections) that can be used at a later time in a replay attack.

One observed trend is that the adoption of existing authentication methods is very slow. After more than ten years, single-factor authentication is still applied at a considerable number of banks, while alternatives have been available. Another trend that we see is that the development and widespread implementation of new authentication methods does not occur often. Since 2002, the only new and widely adopted basic authentication method that we observed is the use of SMS text messages for OTP authentication. All other now widely used authentication methods were already known in 2002.

3. Attack vectors for digital bank fraud

In this section, we map the attack surface of online banking fraud to get an overview of different ways in which online banking can be attacked. There are different ways to conduct fraud in online banking, and the attack surface can be scoped to what is shown in Figure 9.

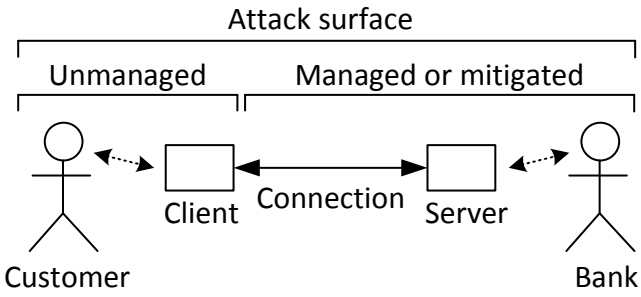

Figure 9: The complete attack surface between customer and bank.

Attacks can be made on actors, electronic devices, the network between electronic devices or a combination of any of these. 'Client' and 'Server' represent the technical infrastructures operated by respectively the customer and the bank. The 'Bank' actor represents the bank employees which have access to critical functions of the technical infrastructure concerning financial transactions.

Parts of the attack surface will be examined further on their vulnerability to attacks. We use public data for our research, which is why our main focus will be on the customer, the customer's technical infrastructure and the connection between the infrastructures of the customer and the bank. We

acknowledge that the bank as a collection of technical and human entities is also part of the attack surface, but we give this less attention due to the limited availability of public information.

3.1. Customer compromise

The customer is the entity that can legitimately manage a bank account. An attacker can choose to impersonate the customer and use their own technical infrastructure to communicate with the bank. One way to gain the required information (credentials) is to apply social engineering.

Social engineering concerns the exploit of human psychology to make victims do or provide what an attacker cannot do or does not have. Through social engineering, it is possible to bypass any technical security precaution. For example, an attacker does not have the necessary information to log in to an account at a bank that applies password authentication. Instead of relying on technical attacks, the attacker can contact the customer, claim that he or she works for the bank of the customer and ask the customer's account number and password using a (for the customer) plausible excuse (technical maintenance, account confirmation, etc.). If the customer gives the information, the social engineering attack is successful. The attacker only has to log in using the provided information and set-up transactions. Social engineering attacks can also be done with more complex authentication methods, such as combinations of OTP and challenge-response authentication. The attacker has to adjust what is told to and asked from the customer for different authentication methods.

Fraud attacks against online banking that rely on social engineering and not on technical exploits are varied. social engineering attacks against online banking have in common is that they attempt to acquire information that allows the attacker to impersonate the victim. The attacker impersonates the victim the moment personal authentication information is used to transfer money from the victim's account to an account of the attacker's choosing.

Getting information with social engineering is often done through an activity known as phishing. With phishing attacks against online banking, the attacker creates a forged message that impersonates the style of a bank. This message is send in bulk to a large number of recipients. The probability that the message will reach potential victims who are customers at the bank chosen by the attacker to forge a message from is raised by sending it in bulk. The forged message contains information that requests the recipient to give sensitive information in the case of customer compromise. This concerns

identification (e.g. a user name or account number) and authentication information (e.g. a password or an OTP) when online banking customers are attacked. One way to get this information from customers is by sending them an e-mail that redirects them to a website under the attacker's control. The phishing site looks similar to the site of the bank. It has an input form and asks the customer to give the relevant information. Note that in this example the technical infrastructure of the customer is not compromised. While the customer's computer is used to visit a phishing site, the attacker only wants information from the customer and does not attempt to access the customer's computer. It is therefore an attack that relies on social engineering. Used phishing channels include (but are not limited to) phone, SMS and email [35].

3.2. Client computer compromise

Another way for an attacker to gain access to the required information to commit fraud attacks is through devices that a bank customer uses for online banking. A client computer is compromised if malicious parties can run their own code on it. Various attacks exist to gain initial entry. This includes a form of social engineering (for example, through phishing) that is different as discussed in the previous subsection. Instead of gaining information from the bank customer, the goal with social engineering for client computer compromise is to let the user open a possibly vulnerable side of the used PC or MD to the attacker. For example, an attacker entices a user to open an attachment containing illegitimate code (possibly hidden in a popular document format) or to visit a site which offers illegitimate code to the device. If the device is vulnerable, the code is executed and the attacker has gained initial entry.

Other approaches for initial entry exist that do not depend on enticing users. An example is 'malvertising', a portmanteau of 'malware' and 'advertising' [36]. Many sites on the Internet use advertisements provided by third-parties as a source of revenue. The used underlying mechanism for embedding third-party content in a web page is often used by attackers to compromise a large number of computers. Instead of compromising a popular site (visited by many potential victims), a less exposed advertisement provider can be used to exploit the user's computer. Cross-site scripting relies on the user to visit a popular, legitimate and trusted site that tries to run illegitimate code on the user's device through the services of a compromised third-party advertisement provider. In this case, the user does not show any unexpected or unwanted behavior.

With a successful attack that compromises a computer, it is possible for an attacker to modify information that a computer receives and that it sends through its input and output interfaces. This includes interfaces for user interaction (e.g. monitor, keyboard and mouse) and network interfaces (e.g. a connection to the Internet). Some form of illegitimate code that runs on the targeted PC or MD is applied in such an attack. This is often referred to as malicious software or malware.

If a customer's bank only requires an account number and password for authentication, it is in most cases enough for an attacker to only log and retrieve a customer's keyboard input. If a bank uses an onscreen keypad (to be used with a mouse cursor or a touch screen) that randomizes the location of the keys in each session or after each press, the attacker can capture the screen output to retrieve the password. A retrieved account number and password (or PIN) can be used by an attacker to log in at the bank at a later time to initiate transactions.

Logging the input of the customer is not always enough to circumvent multi-factor authentication. If, for example, OTP or challenge-response authentication is used, it is not enough for an attacker to passively log the information a customer enters in the client computer because the information given by the customer can only be used once and is worthless once it is received by the bank.

By changing what the customer and what the bank see, it is possible to work around the limitations that multi-factor authentication introduces. Some of the approaches are as follows[23]:

- When the customer visits the site of the bank, the malware modifies the web page that is presented. The information that the customer enters for entity authentication is not send to the bank, but retrieved by the attacker. After the customer has entered the information the bank normally asks and confirmed its input (by choosing 'OK', 'next', 'login' or something similar), a page is shown with a "please wait" prompt. While this prompt is up, the required information for entity authentication is used by the customer's computer (without the customer's knowledge) to start a secure session with the bank and to send a fraudulent transaction.

[23]McAfee, Dissecting Operation High Roller: `http://www.mcafee.com/us/resources/reports/rp-operation-high-roller.pdf`

If more information from the customer is required when transaction authentication is used (e.g. an OTP received by text message or a response with challenge-response authentication), it can be asked directly from the customer, who is still at the 'please wait' prompt. When the required information for transaction authentication is entered, it is used in the session that is hidden from the customer.

The malware stays active afterwards, but allows the customer to log into the bank site. Several characteristics of the pages that the customer sees are changed to hide the fraudulent transactions, such as the total account balance and the transaction history. This conceals the theft long enough for the attacker to make the money untraceable.

- In a variation of the previously described attack, the customer's computer does not connect with the bank. It only connects with the attacker's server. Instead of having the client computer initiate fraudulent transactions, the client computer sends all the required customer authentication information to the server using a fake site of the bank that is presented to the customer. The server establishes a session with the bank and creates one or more fraudulent transactions with the retrieved information. The customer is given a prompt that states that the system is under maintenance and that the customer should retry again after a number of hours or days. This gives the attacker enough time to transfer the money further. The customer is allowed to login again after the fraudulent transactions are finished.

- Another attack method is the use of 'transaction poisoning'. When a customer creates a session with his or her bank, the malware waits until the customer initiates a transaction. Before the transaction is send, critical values are altered by the malware (such as the destination account number). This allows the attacker to circumvent bank controls that verify the amount of a transaction or a set of transactions with a customer before it is accepted.

An important element in these examples is that the customer has an active role in the attack. Unknowingly, the customer authorizes transactions that are prepared by the attacker.

3.3. In-transit compromise

Instead of concentrating on the customer or the computer used by the customer, it is also possible to concentrate on data in-transit. As stated in Section 2.4 (starting on page 17), almost all banks apply SSL/TLS to protect the confidentiality and integrity of the information flow between the bank's and the customer's computers. Several implementation issues exist, but SSL/TLS provides decent protection once a connection is established.

In-transit compromise requires that the attacker has control of the connection or at least part of the connection between both computers. By either exploiting the weaknesses in used SSL/TLS implementations or circumventing the protocol altogether, it is possible to change the traffic between customer and bank. Exploitable weaknesses in SSL/TLS, to which some banks today are vulnerable, are already mentioned in Section 2.4.

Several attacks are possible to circumvent SSL/TLS, although we had trouble in finding examples of applied attacks on online banking. This can be explained by that other methods that compromise the customer or the customer's computer are easier to execute. In-transit compromise requires real-time intervention and is hard to do on a large scale. The opposite is true for the other types of compromise, which have more time for preparation and are often less critical in the timing of the attack.

Assuming that an attacker has full control of the customer's connection, one way to circumvent SSL/TLS is by not offering the site from the bank over SSL/TLS in the first place. When a browser visits a site without explicitly providing the protocol, HTTP is used by default instead of HTTPS. Many bank sites redirect the user to a HTTPS site as soon as the user visits the site over HTTP or when the user chooses to log in into the secure banking environment. If a HTTP site is visited first, the retrieved page can be changed by an intermediate party to not offer HTTPS at all through links on the page. If the user tries to log into the secure environment, the attacker forwards (and modifies, if required for the attack) the secure pages from the bank over an insecure connection to the client computer. This allows the attacker to inject code that asks the customer for authentication details or to add a hidden transaction to a set that is to be authenticated.

One possible mitigation is the use of HTTP Strict Transport Security (HSTS).[24] With HSTS, the first visit to a site is insecure. In the insecure

[24]RFC 6797, HTTP Strict Transport Security: `http://tools.ietf.org/html/`

session, it is established (using an extra HTTP field) that the browser should connect to the secure site on subsequent visits. Once a browser receives the extra HTTP field, it stores the visited site in a local list of sites that must only be visited using HTTPS. The next time the user visits the site, an HTTPS connection will be used even if it is not explicitly stated.

As stated, the first visit is insecure. To mitigate this, some web browsers come 'preloaded' with list of sites that should always be visited using HTTPS instead of HTTP after the browser is installed or updated.[25] The browsers that include such a list will connect with these known sites securely when the user visits them (also the first time).

A test on the banks in our survey shows that none of them support HSTS. The same test on the list of preloaded browsers of Chrome[26] shows that 129 out of 439 sites[27] support HSTS. Google's list includes 3 bank sites that do support HSTS, from which it can be concluded that bank sites rarely support this. This is not really surprising: HSTS is new (it was proposed as a standard in November 2012) and needs some time to be more widely adopted.

Another way to circumvent SSL/TLS is by using an illegitimate but valid certificate. The attack starts with a secure connection between the customer's computer and the attackers intervening node. The attacker requires a certificate signed by a trusted certificate authority in name of the bank. This does not have to be the certificate authority used by the bank. The only requirement is that it is a certificate authority that is trusted by the customer's browser.

It is required to gain a valid certificate for the domain name of a specific bank that is not expired and signed by a trusted party. Leavitt [37] describes an attacker that tried to get access to three certificate authority systems. He succeeded in gaining access to two (one of the systems was from the infamous DigiNotar organization) and used this to create fake certificates.

`rfc6797`

[25]Google Chrome and Mozilla Firefox both include filled a list of hosts that should only be visited using HTTPS in their installations: `https://blog.mozilla.org/security/2012/11/01/preloading-hsts/`

[26]Google Chrome's list of hosts for which HTTPS is enforced: `http://src.chromium.org/viewvc/chrome/trunk/src/net/http/transport_security_state_static.json`

[27]30 sites gave an error and therefore could not be examined. These were excluded in the statistics.

These certificates were valid from a technical perspective (signed by a trusted party, not expired) when used to impersonate services such as Google Gmail.

Another possible direction for an attack is to exploit faulty SSL/TLS implementations on the client-side of the connection. An example is given by a bank that implemented SSL/TLS incorrectly on their mobile application [38]. When a server identifies itself with a certificate, the name on the provided certificate must be checked on whether it matches that of the server (its DNS name), the expiration date must be checked to see if the certificate is not expired and the party that digitally signed the certificate must be trusted by the client before it received the certificate. If any of these checks fails, the certificate must be assumed to be invalid. The bank noted by Houtenbos et al. developed a mobile banking application for the Android mobile operating system. SSL/TLS was implemented for a secure connection between mobile device and bank. Due to a programming oversight, the application did not check received certificates by their name. Any trusted certificate that was not expired was accepted by the application and assumed to be representing the bank.

We are unaware of any cases in which a fake but valid certificate or a faulty SSL/TLS implementation was used to intercept and modify bank traffic in real time. Compared to the other discussed attack vectors, in-transit compromise seems quite cumbersome due to the large amount of required resources.

3.4. Bank-side compromise

As explained at the beginning of this section, we focus our attention on the compromise of the customer, the customer's devices and the connection between the customer's devices and the bank. We still have to acknowledge the bank as an entity, consisting of both a technical infrastructure and its users, which are both part of the attack surface (see Figure 9 on page 35).

Internal (assisted) attacks can occur. With these type of attacks, bank employees commit illegitimate actions as criminals [39] or victims of blackmail [40] for the purpose of fraud. It is difficult to actively defend against attacks that come from the inside, since the mechanisms that are entrusted to bank employees for their everyday work are the same mechanisms that can be used for fraud. Reactive defense is possible by measuring the actions that a user commits to create a baseline of normal behavior. Exceptional behavior at a later time can be a reason for investigation. Fraud will already be committed by the time it is detected with reactive defense, which is why

in an ideal situation actions made by users can be reversed in the time span that is required to detect fraud.

External attacks that aim directly at banks and their infrastructure are also not unheard of. An example is a case in which a distributed denial-of-service (DDoS) attack was applied as a decoy for the compromise of wire payment switches. During the attack, attackers transferred money illegitimately [41]. This use of a DDoS attack is distinct from the use by political activists. In terms of information security, political activists usually only apply DDoS attacks to influence the availability of a service. However, the goal of the DDoS attack in the example was to compromise the integrity of the system for financial gain by acting as a diversion. The DDoS attack itself only influenced availability, but it gave the attackers room to attack the wire payment switches. How the wire payment switches were compromised is unknown. Attack methods that target the customers of banks or the technical infrastructure customers use (through social engineering, malware, etc.) can also be applied on employees and the technical infrastructure of banks.

Another attack vector is represented by possibly existing backdoors in the technical infrastructure used by banks. Products and services that banks use can contain undocumented features or artificial-made vulnerabilities that allow third-parties to compromise the information flow. Recently, the United States National Security Agency (NSA) has been in the news headlines due to a large leak of documents. It is reported that some of these documents state that the NSA is directly or indirectly responsible for the inclusion of vulnerabilities in commercial software [42]. While this is speculation, it at least tells us that the deliberate inclusion of hidden vulnerabilities in commercial software by third-parties is not inconceivable.

4. Open security issues in online banking

Several approaches for online bank fraud attacks are described in the previous section. In this section, we note security issues that are exploited today. By describing these issues, possible improvements can become more tangible. The main focus of our research is on the customer-side and the connection between customer and bank, and not on the bank or its infrastructure. Therefore, the noted issues will relate to online bank fraud that concerns the compromise of the customer or the technical infrastructure used by the customer.

Multi-factor authentication has seen been adopted more widely for online banking between 2002 and 2013 (based on our survey, as noted in Section 2), but did not reduce the vulnerability of the customer or the customer's technical infrastructure as parts of the attack surface (as noted in Section 3). We will first describe what multi-factor authentication did not solve before we describe what is currently missing in online banking authentication. Another issue that we note is the effective man-in-the-middle between customer and bank that is represented by the customer's PC or MD. We conclude this section with an overview of the problems that these security issues contribute to.

4.1. The role of multi-factor authentication

Multi-factor authentication prevents one specific attack method. This is the retrieval of the customer's authentication credentials through either social engineering or installed malware on the computer used by the customer (keystroke logging)[28] and use of the authentication credentials at a later time in a replay attack. When an attacker uses a retrieved password to establish a session with a bank, it is a replay attack since the same authentication credential is used and accepted multiple times.

With multi-factor authentication applied in online banking, only retrieving the customer's input is not enough for a successful attack. Each encountered multi-factor authentication solution in our survey adds an element that prevents replay attacks. It is unlikely that an OTP will be used twice and if it does, the bank can simply block the log in attempt by registering previously used OTPs. With challenge-response authentication and digital signature generation, it is unlikely that the same challenge will be given twice. Retrieving an OTP or a response is useless when they are already received by a bank, since they cannot be used again.

Retrieval of authentication credentials without active participation of the customer is all multi-factor authentication protects against, and this is not enough. Social engineering and malware are methods that are still available to attackers. See Section 3.1 (page 36) and Section 3.2 (page 37) for more information on how social engineering and malware are applied to circumvent multi-factor authentication.

[28] SSL/TLS protects passwords adequately in-transit, preventing eavesdroppers from obtaining it in a passive attack.

4.2. A lack of proper authentication

The majority of banks in our research use multiple factors to authenticate the customer. This mitigates the problem of passive attacks, such as eavesdropping and password guessing. Multi-factor authentication does not provide protection against active attacks (attacks that involve the banking customer) and therefore does not provide adequate protection against fraud [43].

What is currently authenticated in online banking is the customer and not the customer's actions. This also includes attack scenarios in which multi-factor authentication is applied: the attacker requires (recent and unused) authentication information from the customer to create fraudulent transactions. A description of customer and transaction authentication was given in Section 2.5. Some banks apply transaction authentication by providing (physical) or letting the customer generate (electronic) one-time passwords to be sent to the bank (see Section 2.5.2). While this does provide more reassurance that it is the customer that is authenticating (due to the addition of the possession-factor), it does not represent the transactions themselves. The same OTP can be used for both legitimate and fraudulent transactions.

Other banks apply challenge/response authentication for transaction authentication (also discussed in Section 2.5.2). The same problem is present if information of each transaction is not part of the challenge in a human-readable format. Again, the customer is authenticated and not the transactions. Even if information from all transactions is part of the challenge, it does not help the customer in identifying fraudulent transactions if the information is aggregated in a short numerical string. Some banks mitigate this by making certain characteristics of a set of transactions part of the challenge. An example is the rounded down total sum of all transactions. This is not a full mitigation, since it is still possible for an attacker to add an extra transaction that takes part of the sum of the other transactions. Alternatively, the attacker can simply change a destination account number of one or more transactions. Transaction set authentication does not provide complete transaction authentication.

There are also banks that do not apply multi-factor authentication at all. This applies to most banks in the United States, which only require a username and password when logging in. The explained motivation is that most fraud transfers are reversible and that actual losses by banks are too limited to implement alternative authentication methods. Retrieving passwords is not the bottleneck in fraudulent attacks. Instead, the restrictions given by

banks on transactions that are irreversible and untraceable limit the modus operandi of attackers. If money mules[29] are used, they pay for the damages, not the bank nor the victims. What can be considered the lower bound in terms of security (passwords) is compensated by banks through the upper bound of protection (zero liability towards victims) [44, 45].

Instead of concentrating primarily on avoiding or mitigating the misuse of personal information, banks must apply more effort in preventing the creation of fraudulent transactions. Customers do not create fraudulent transactions, despite that the required information is given to or intercepted by an attacker to create them. Customers cannot be involved in any solution because they are not involved with fraudulent transactions in the first place [46].

If banks would verify the authenticity of a transaction by checking that (a part of[30]) the destination account number, the currency and the sum of each transaction are provided by the customer, real transaction authentication would take place. It is not interesting for an attacker to change information from a transaction for monetary gain if these fields are verified and cannot be changed.

An account holder is the only party that can create legitimate transactions, aside from transactions that are a consequence of legal action. Therefore, the account holder is the only party that can genuinely distinguish legitimate transactions from fraudulent transactions. Certainty that from a transaction the destination account number, the currency and the sum can only be provided by the customer is not enough to non-repudiate the entire transaction, but it is enough to non-repudiate the financial assets associated with the transaction. Transaction authentication must secure the integrity of these fields for each transaction to avoid third-party financial fraud that does not involve the customer.

4.3. The persistent man-in-the-middle

With online banking as it is today, the presence of a persistent man-in-the-middle cannot be avoided. The persistent man-in-the-middle is represented by everything that is not the customer and that of which the security cannot

[29] A money mule is a party that (temporarily) gives an attacker access to their bank account. The account is a recipient of illegitimate transactions and used to withdraw money associated with these transactions (irreversibly and untraceable) as cash.

[30] Assuming that account numbers are distributed randomly, only checking part of an account number would provide reasonable security against changed account numbers.

adequately be managed or mitigated by the bank. Referring to Figure 9 on page 35, the part of the attack surface that conforms to this description is the client computer used by the customer. While the connection between customer and online bank is not managed by the bank when it uses the Internet, we note that confidentiality and integrity are well enough provided by SSL/TLS (see Section 3.3 on page 40 for more information).

Between the SSL/TLS connection and the customer is a client computer (PC or MD). The customer does not have any interaction with what can be labeled as 'the bank' aside from any authentication devices offered by the bank. For online banking, the customer interacts with a client computer which has far more functionality than that is required for only online banking and which is not managed by the bank. The bank tells the client how to present information to the customer and how to get information from the customer to the bank. Other parties aside from the bank and the customer can also influence the client (a man-in-the-middle attack) and therefore interfere in the exchange of information. A bank might be able to check how the client responds and judge whether it has been compromised or not, but attackers only have to change how the client responds to continue the cat-and-mouse game between detection and avoiding detection of a compromised client computer. Attackers have more influence on the systems used by customers compared to banks, since they have the first-move advantage and are not restrained by legitimacy or morality.

4.4. Problems and solvability

The following (interrelated) problems can be defined:

- Financial fraud. Assuming that the primary motivation of an attacker is financial gain, the problem exists that attacks on the customer or on the technical infrastructure of the customer have the goal of transferring funds illegitimately.

- Privacy infringement. An attacker can have access to information such as transaction history, credit card bills, an account's sum, etc. While this problem is inconvenient and possibly embarrassing for a customer, it also gives the attacker more information for subsequent social engineering attacks.

The lack of proper authentication relates to financial fraud, while the persistent man-in-the-middle relates to both. Security improvements in online banking need to solve the issues that are the cause of the problems.

It is difficult to solve the privacy problem. Due to the persistent man-in-the-middle, there is no guarantee that there is not an eavesdropper listening in on the communication between customer and bank or abusing the customer's session on a PC or MD to gain more information. From a cost and usability perspective, it is unlikely that a bank can create a completely secure communication path between customer and the bank's infrastructure in the near future.

The threshold for solving the financial problem is lower. A bank can apply reactive detection of fraudulent transactions based on baseline criteria from previous transactions to stop these transactions before money is lost. But this does not have to be the only defense. While it is not cost-effective nor user friendly to distribute a complete technical infrastructure for online banking solutions to customers, it is possible to distribute electronic authentication devices as an active protection measure (as proven by many banks today). If such a device can be used to not only authenticate the customer (as done today) but also the customer's intent, it is possible for the bank to act on actions done by the customer instead of on the behavior of an attacker. A customer must (for example) decide where which amount of money goes to from his or her bank account. While the decision is made by the customer, it is up to the bank to clarify the information on which a decision is based on and the consequences of the decision before it is made by the customer.

5. Concluding remarks

The security issues that we discussed relate to the issues as discussed by Claessens et al. in [1]: the establishment of a secure channel for data confidentiality and integrity, and the authentication of the customer (entity authentication) and each transaction (transaction authentication).

Claessens et al. concluded for the first issue that SSL/TLS was used by most banks and proven to be a practically secure protocol from a cryptographic point of view. While we have noted several issues in SSL/TLS as it is used in today's online banking, this conclusion still stands due to that exploiting weaknesses in SSL/TLS communication requires a vast amount of resources to attack a limited number of targets compared to other discussed approaches (see Section 3.3 on page 40 for more information).

The other attack vectors are the reason why the technical infrastructure for communication security must not be scoped as only the connection between customer and bank devices. The devices have to be included, for they

are also (and more, compared to SSL/TLS) vulnerable to attacks that challenge confidentiality and integrity of the communication between customer and bank. In 2002, Claessens et al. noted that the best security solution for online banking relies on the assumption that the end-points of the system are trusted. In our article, we noted the persistent man-in-the-middle, represented by the client computer used by the customer. A bank cannot trust the client computer as an end-point since it cannot provide a trusted and guaranteed untampered environment for online banking.

Banks take advantage of web standards to reach a large number of online banking customers that use PCs and MDs. Many banks also provide online banking applications for MDs. This use of customer-owned PCs and MDs must be considered insecure by banks. All client-side code (including web browsers and banking applications) is executed in an environment that cannot guarantee confidentiality or integrity, because third-parties can influence what customers and banks see through PCs and MDs (see Section 3.2 on page 37 for more information). The client computer cannot be removed from online banking since it is an important aspect that allows banks to electronically reach large audiences. Therefore, the current best practice is that banks do not rely on client computers for entity and transaction authentication to prevent fraud attacks.

As their second issue, Claessens et al. concluded that fixed passwords were the most widely used method for authentication. It was noted that the best practice at the time was probably the use of hardware tokens or the use of a smart card for challenge-response authentication and creating digital signatures. One third of the banks in our survey apply the best practices from 2002 in 2013. The majority that does not apply the best practices still implement some form of multi-factor authentication (see Section 2.5 on page 25 for more information). While the use of only fixed passwords for authentication with PCs is still seen at almost a quarter of all examined banks, the majority of banks offer some form of multi-factor authentication.

However, multi-factor authentication is not enough. Current implementations of it do confirm customer identity, but not intention. Actions themselves and the results of actions of the customer do not have to conform with the intention of the customer. Customer can be tricked through social engineering to do something that they do not want. Also, if a customer initiates an action with genuine intent, it is possible for an attacker through the persistent man-in-the-middle (the customer's device) to modify the resulting information from that action before a bank receives it. It is not always

possible to differentiate between both types of attack due to the lack of non-repudiation of the customer's actions. To get more insight in financial fraud, it is required to non-repudiate bank transactions. This will not prevent fraud completely since it is still possible through social engineering. However, it does prevent that a party other than the customer changes the outcome of the customer's actions (such as by changing a destination account number or the sum of a transaction).

Similar to Claessens et al. we close our work by noting that perfect security cannot be achieved. Online bank fraud will most likely always exist in some form as long as online banking exists. However, there is room for improvement on how security threats and opportunities are handled.

Banks have shown effort in attempts to improve security, but overall security has not increased. This is because security is often an afterthought after something is implemented. Efforts therefore often go towards fixing existing problems in newly offered functionality. Putting the right amount of effort at the right time in securing the vulnerable parts of the attack surface that need it can reduce the impact of both existing and new attack methods. Determining underlying (potential) issues based on the symptoms should be given proper attention before steps towards improvements are taken.

Improving security is not something that only has to be done once. A continuing process is required that monitors and categorizes threats and opportunities. Security must be given the same priority as functionality and not tag along as a nuisance that is based on only the problems that were relevant a decade ago.

6. Acknowledgements

We thank Wouter Stol from Open Universiteit for his thorough proof reading and advice. Our thanks also go out to our colleagues at Radboud University who assisted with extracting, translating and verifying information from bank documentation.

Bibliography

[1] Joris Claessens, Valentin Dem, Danny De Cock, Bart Preneel, and Joos Vandewalle. On the security of today's online electronic banking systems. *Computers & Security*, 21(3):253–265, 2002.

[2] Google. Google Play (last retrieved: 2014-01-16), 2014. URL https://play.google.com/.

[3] Apple. Apple's App Store (last retrieved: 2014-01-16), 2014. URL https://itunes.apple.com/en/genre/mobile-software-applications/id36.

[4] Google. Android Developers - Security Tips (last retrieved: 2013-11-21), 2013. URL http://developer.android.com/training/articles/security-tips.html.

[5] Apple. ioS App Programming Guide: The iOS Environment (last retrieved: 2013-11-21), 2013. URL https://developer.apple.com/library/ios/documentation/iphone/conceptual/iphoneosprogrammingguide/TheiOSEnvironment/TheiOSEnvironment.html.

[6] Steve Mansfield-Devine. Android architecture: attacking the weak points. *Network Security*, 2012(10):5–12, 2012.

[7] Arron Hirst. Apple: 93% Of Customers Are Using iOS 6 (last retrieved: 2013-11-21), June 2013. URL http://www.razorianfly.com/2013/06/21/apple-93-of-customers-are-using-ios-6-chart/.

[8] Roland Van Rijswijk-Deij and Erik Poll. Using trusted execution environments in two-factor authentication: comparing approaches, 2013. Unpublished results by researchers of Radboud University Nijmegen (the Netherlands), Faculty of Science, Digital Security.

[9] Alan Freier, Philip Karlton, and Paul Kocher. RFC 6101, describing SSL 3.0 (last retrieved: 2014-01-16), August 2011. URL http://tools.ietf.org/html/rfc6101.

[10] Tim Dierks and Christopher Allen. RFC 2246, describing TLS 1.0 (last retrieved: 2014-01-16), January 1999. URL http://tools.ietf.org/html/rfc2246.

[11] Tim Dierks and Eric Rescorla. RFC 4346, describing TLS 1.1 (last retrieved: 2014-01-16), April 2006. URL http://tools.ietf.org/html/rfc4346.

[12] Tim Dierks and Eric Rescorla. RFC 5246, describing TLS 1.2 (last retrieved: 2014-01-16), August 2008. URL http://tools.ietf.org/html/rfc5246.

[13] Sean Turner and Tim Polk. RFC 6176, describing how TLS must not be backwards compatible with SSL 2.0 (last retrieved: 2014-01-16), March 2011. URL http://tools.ietf.org/html/rfc6176.

[14] WAP Forum. WAP WTLS Specification Version 06-Apr-2001 (last retrieved: 2013-11-20), April 2001. URL http://technical.openmobilealliance.org/tech/affiliates/wap/wap-261-wtls-20010406-a.pdf.

[15] Qualys. Qualys SSL Labs SSL Server Test (last retrieved: 2014-01-16), 2013. URL https://www.ssllabs.com/ssltest/.

[16] COMODO. COMODO SSL Analyzer (last retrieved: 2014-01-16), 2013. URL https://sslanalyzer.comodoca.com/.

[17] Ron Rivest. RFC 1321 - The MD5 Message-Digest Algorithm (last retrieved: 2014-01-16), April 1992. URL http://tools.ietf.org/html/rfc1321.

[18] CERT. CERT Vulnerability Notes Database - MD5 vulnerable to collision attacks (last retrieved: 2013-10-14), December 2008. URL http://www.kb.cert.org/vuls/id/836068.

[19] Xiaoyun Wang and Hongbo Yu. How to break MD5 and other hash functions. In *Advances in Cryptology–EUROCRYPT 2005*, pages 19–35. Springer, 2005.

[20] Tao Xie and Dengguo Feng. How to find weak input differences for MD5 collision attacks. *IACR Cryptology ePrint Archive*, 2009:223, 2009.

[21] Hans Dobbertin. The status of MD5 after a recent attack. *CryptoBytes*, 2(2), 1996.

[22] Vincent Rijmen, Florian Mendel, Norbert Pramstaller, and Christian Rechberger. Current status of SHA-1 (last retrieved: 2013-11-21), February 2007. URL https://www.signatur.rtr.at/repository/rtr-sha1-20070221-en.pdf.

[23] Elaine Barker and Allen Roginsky. Transitions: Recommendation for transitioning the use of cryptographic algorithms and key lengths. *NIST Special Publication*, 800:131A, 2011.

[24] Elaine Barker, William Barker, William Burr, William Polk, and Miles Smid. Recommendation for key management–part 1: General (revision 3). *NIST Special Publication*, 800:57, 2012.

[25] Qualys. SSL/TLS Deployment Best Practices (last retrieved: 2013-11-21), February 2012. URL `https://www.ssllabs.com/downloads/SSL_TLS_Deployment_Best_Practices_1.0.pdf`.

[26] Thai Duong and Juliano Rizzo. Here come the ⊕ ninjas (last retrieved: 2013-11-21), May 2011. URL `http://nerdoholic.org/uploads/dergln/beast_part2/ssl_jun21.pdf`.

[27] NIST. National Vulnerability Database (CVE-2011-3389) (last retrieved: 2014-01-16), October 2013. URL `http://web.nvd.nist.gov/view/vuln/detail?vulnId=CVE-2011-3389`.

[28] Qualys. CRIME: Information Leakage Attack against SSL/TLS (last retrieved: 2013-10-14), September 2012. URL `https://community.qualys.com/blogs/securitylabs/2012/09/14/crime-information-leakage-attack-against-ssltls`.

[29] Nadhem AlFardan and Kenny Paterson. Lucky Thirteen attack (last retrieved: 2013-11-21), February 2013. URL `http://www.isg.rhul.ac.uk/tls/Lucky13.html`.

[30] CA/Browser Forum. Guidelines For The Issuance And Management Of Extended Validation Certificates (last retrieved: 2013-11-21), May 2012. URL `https://www.cabforum.org/Guidelines_v1_4.pdf`.

[31] Collin Jackson, Daniel R Simon, Desney Tan, and Adam Barth. An evaluation of extended validation and picture-in-picture phishing attacks. In *Financial Cryptography and Data Security*, pages 281–293. Springer, 2007.

[32] Jennifer Sobey, Robert Biddle, Paul Van Oorschot, and Andrew Patrick. Exploring user reactions to new browser cues for extended validation certificates. In *Proceedings of the 13th European Symposium on Research*

in Computer Security: Computer Security, ESORICS '08, pages 411–427, Berlin, Heidelberg, 2008. Springer-Verlag. ISBN 978-3-540-88312-8. doi: 10.1007/978-3-540-88313-5_27. URL http://dx.doi.org/10.1007/978-3-540-88313-5_27.

[33] Robert Biddle, Paul Van Oorschot, Andrew Patrick, Jennifer Sobey, and Tara Whalen. Browser interfaces and extended validation SSL certificates: an empirical study. In *Proceedings of the 2009 ACM workshop on cloud computing security*, pages 19–30. ACM, 2009.

[34] Christina Braz and Jean-Marc Robert. Security and usability: the case of the user authentication methods. In *Proceedings of the 18th International Conference of the Association Francophone d'Interaction Homme-Machine*, pages 199–203. ACM, 2006.

[35] SCAMwatch. Requests for your account information (last retrieved: 2013-10-14), 2006. URL http://www.scamwatch.gov.au/content/index.phtml/tag/requestsforyouraccountinformation.

[36] Aditya Sood and Richard Enbody. Malvertising–exploiting web advertising. *Computer Fraud & Security*, 2011(4):11–16, 2011.

[37] Neal Leavitt. Internet security under attack: The undermining of digital certificates. *Computer*, 44(12):17–20, 2011.

[38] Thijs Houtenbos, Jurgen Kloosterman, Bas Vlaszaty, and Bas de Koning. Security in mobile banking, 2012. Unpublished results by students of the University of Amsterdam (the Netherlands), System & Network Engineering.

[39] Brett Warfield. Employee Fraud in Australian Financial Institutions (last retrieved: 2014-01-16), November 2013. URL http://www.warfield.com.au/Warfield_Employee_Fraud_in_Australian_Financial_Institutions.pdf.

[40] Cumbria Constabulary. Kendal pensioner sentenced for blackmailing Milnthorpe bank (last retrieved: 2014-01-16), February 2013. URL http://www.cumbria.police.uk/news/latest-news/kendal-pensioner-sentenced-for-blackmailing-milnthorpe-bank.

[41] SC Magazine. Fraudsters target "wire payment switch" at banks to steal millions (last retrieved: 2013-10-14), August 2013. URL `http://www.scmagazine.com/article/307755/`.

[42] The Guardian. 'How US and UK spy agencies defeat internet privacy and security' (last retrieved: 2013-10-14), September 2013. URL `http://www.theguardian.com/world/2013/sep/05/nsa-gchq-encryption-codes-security`.

[43] Bruce Schneier. Two-factor authentication: too little, too late. *Communications of the ACM*, 48(4):136, 2005.

[44] Cormac Herley, Paul Van Oorschot, and Andrew Patrick. Passwords: If we're so smart, why are we still using them? In *Financial Cryptography and Data Security*, pages 230–237. Springer, 2009.

[45] Dinei Florencio and Cormac Herley. Is everything we know about password-stealing wrong? *Security & Privacy*, 10(6):63–69, 2012.

[46] Bruce Schneier. *Schneier on Security*, pages 205–208. Wiley, 2005.