RP2
Online Banking: Attacks & Defences

University of Amsterdam
Master of Science in System and Network Engineering
Class of 2008-2009

Dominic van den Ende (dominic.vandenende@os3.nl),
Tom Hendrickx (tom.hendrickx@os3.nl)

July 3, 2009

Abstract

This paper gives a presentation of user and transaction authentication models in the banking world. It demonstrates that most of the currently used authentication models are not up to the task of ensuring a secure online banking experience, because of the latest forms of trojans. This is why, new methods are being examined and forged into new models which must ensure a more adequate level of security for online transactions.
This project has been supported by a corporate research partner who has provided us with reference data on fraud and anti-fraud measures, on the condition that we guarantee his anonimity.

Contents

1 Introduction

In the world of online banking there has always been a constant struggle between frauds and malware writers, and the security community. Since the security community introduced two-factor authentication, online banking was deemed relatively safe for quite some time. New types of attack have risen, however, and brought this relative safety to an end. Currently, one of the most popular attacks is what is called a Man-in-the-Browser (MitB) attack. In such an attack, the web browser is hijacked and sits more or less in between the banking-application and the web browser. As such, it is possible to show the correct data to the end user, whilst sending fake data to the banking site. Data like bank account number or amount could be changed, all in favour of the attacker.

This means that the client side can not be fully trusted. To a large extent because the client side does not reside in a fully controlled environment, such as on the server side. In other words, if an adequate level of security needs to be obtained, a one sided solution will not suffice.
This problem has lead to this research project, in which we have considered a multi-level security approach for online banking, consisting of the following layers:

Layer 1: Security of the end user PC

Layer 2: Extra out-of-band authentication

Layer 3: Back-office monitoring

We have especially focused on two different parts of online banking. On one hand there is the user/client authentication when logging in and on the other there is transaction authentication. The goal is to fend off known issues like shoulder surfing, identity theft, phishing, Man-in-the-Middle, Man-in-the-Browser, etc (Appendix A). This brought us to the following research questions:

- Examine the current used models of authentication and consider their strengths and weaknesses.
- Which methods can be used in one of the three different security layers and compare them on points such as maturity, potential and effectivity.
- Propose new models, based on known elements in combination with the newly found methods for a more secure level of authentication, whilst considering the level of user convenience and the costs per user.
- Make a proposition of a balanced model and analyse this architecture against current trojans and speculate how future trojans may evolve if confronted with this new architecture.

This report has the following structure. Firstly, the currently used models of authentication are described, together with their strengths and weaknesses. Section 3 then presents possible methods which can be used to counteract some of the weaknesses discussed later on. This is followed by a presentation of our next generation of models in section 4. These are mostly based on existing models in combination with some of the new methods, to offer the best possible protection. Based on one of these models, a prediction is then made of possible future trojans in section 5. Section 6 concludes this report.

2 Current models

In terms of authentication methods, the online banking world can be divided into two main groups. One which is convinced of the necessity of two-factor authentication and one which is not. Examples of both are presented in this section.

Multi-factor authentication must consist of at least two of the following factors:

1. Something you know (e.g. a password, PIN, . . .)
2. Something you have (e.g. a scratch list or a device such as a token, a mobile phone, . . .)
3. Something you are (e.g. biometric data)

An important consideration of the second factor is that it can not be easily faked or copied.

2.1 Models without two-factor authentication

Since the end of 2006, US financial institutions have been obliged by the Federal Financial Institutions Examination Council (FFIEC) to implement two-factor authentication [1]. US culture has, however, a predefined love for user convenience. As a consequence this has counteracted the implementation of proper two-factor authentication methods. Nevertheless several US banks advertise the statement of protecting their online banking with two-factor to even three-factor authentication. This is why we have chosen two of their current models as an example to show their ineffectiveness.

Our first example comes from Northwestern Bank, which states to be using two-factor authentication [2].

1. Username and password, which is something you know.
2. A secure cookie, which is supposed to be something you have.

Whilst the first factor is interpreted correctly, the second is a false statement. First of all, something you have is meant to be a secundary device, which must unable any false authentication if the computer is compromised. Secondly, the ”secure cookie” is meant to make the authentication proces even easier, not more secure. It works as follows: A user who wants to authenticate on a website, may choose to ”add extra protection” to that computer. The first time and everytime if the secure cookie is not used, an extra question, which only the user would be able to answer, will be asked. The cookie contains information of the used computer and the user account of the person who is logging in. This has two consequences. First of all, if the user choses the extra protection option, an attacker only needs to shoulder surf or compromise the computer with a simple keylogger to acquire the username and password. With this information, he will be able to abuse the account from that computer, because the cookie has flagged

the computer as safe for the banksite. If the attacker wishes to abuse the account from a remote location, he can trigger the banksite to ask the extra question by deleting the secure cookie. With the keylogger, which is in place, the answer to this question will also be logged. Depending on the number of possible questions, the attacker can choose between two options. He can choose to delete the cookie for a number of times until he has the answers to all possible questions or he can retry the authentication process from the remote computer until the question appears, to which he knows the answer. Of course he enables the "extra protection" option, which enables him to log in each subsequent time only needing the username and password.
This is a clear example of the misuse of the term "two-factor", as it can be reduced to something you know both times. Furthermore, because only a keylogger is needed for a successful fraudulent transaction, the level of security of this model can be regarded as very low.

Our second example comes from the Pacificcrest Bank, which even states to be using three-factor authentication [3].

1. A password, which is something you know.

2. A security key in the form of a graphical representation, which must always be present on the site to prove that its not being spoofed, is supposed to be something you have.

3. A question out of three to which only you know the answer, is suggested to be identifying information and is supposed to be something you are.

Again, it is clear that this is a model where only one-factor authentication is used, but in threefold. All factors can be taken back to something you know. Strangely enough, the second step, a "reverse authentication" (the bank identifies itself to the client), is suggested to be a third factor. Furtheron, the first and the third step need an answer, which can be obtained by an attacker with a simple keylogger. Knowing them is enough to enable the attacker to remotely log in with the victim's credentials, because the second step is only a form of authentication from the banksite to the user. This makes spoofing of the site to obtain data from the first and the third step more difficult, but it is not impossible because the picture can be obtained from the victim's temporary internet files, eventhough, the user must have visited the banksite from that computer and the attacker needs to have compromised the computer with malware to obtain the picture. It can be done even more easily, however.
If the user can be lured to a spoofed site, there are a few methods to trick the user in believing it to be the real site. First, if the user logs in, it is possible to slow down the loading of the second page. Meanwhile, after the spoofed site has made a connection to the real banksite with the user's credentials and it shows the picture, the second page can be loaded for the user with the obtained picture.
Secondly, if the user has sent his credentials to the spoofed site, that site can easily give a failure message and redirect the user to the real banking site. Depending on the user's

awareness, the attack may never be noticed.

The easiness in both models to obtain the user's credentials clearly proves the inadequacy of one-factor authentication as a sufficient security measure.

2.2 Models with two-factor authentication

The most frequently used two-factor authentication exists of combining something you know with something you have. The latter might be a scratch list, a (mobile) phone, a one-time token or a short-time token. Its use can also be different from model to model. Models can be based on a secondary factor for user authentication only, for transaction authentication only or for both. As far as we have encountered, only the latter two are being used in the banking world today.

User authentication scam
Some models are based on the fact that only the making of transactions needs extra protection. This means that the possibility of viewing all previously made transactions and the current balance of the accounts by a fraud is deemed to be less risky. Aside from the danger of easily exposing privacy sensitive information, it has been misused in a certain scam, where a security issue was used to replay the customer's authentication. This enabled them to view the customer details and historic transactions, which was enough to proceed with the scam. This was on account of some online services which allow a bank account to be authorised for future random "direct debets", based on the fact that you correctly confirm to payments from the same online service. The fraudster's owned online service account can this way be linked to an unaware user, who's account was linked based on replaying his/her user authentication.

This comes close to the dangers of a model without two-factor authentication. An attacker only has to retrieve the user's login and password, by means of a keylogger or by shoulder surfing to make this scam possible. This is why we strongly suggest to not only use two-factor authentication for transaction authentication, but also for user authentication. This makes simply obtaining the user's credentials insufficient to an attacker to log in or make a transaction with information that might have been intercepted, because the entered codes are unique for each login and transaction. As a consequence, the danger of any attack aimed at stealing a user's credentials, such as shoulder surfing, phishing, pharming, etc. has been neutralised.

Man-in-the-Browser attack
This is also where a new form of fraud has been implemented, the Man-in-the-Browser attack. As stated before, this attack can change the data which is presented by the browser's GUI, because it uses HTML injection, and the data which is transmitted to the banking site, because it circumvents the SSL session such that the confidentiality and integrity is lost . This makes all previous countermeasures useless. What is presented

Only one technique, which is currently being used by several banks, has been successful in countering this attack. This is being done by the use of out-of-band transaction control in addition of two-factor authentication. The two methods that are currently used for out-of-band transaction control are:

- SMS message with transaction information plus a transaction authentication (TAN) code, which is for example used in the ING online banking model [4].
- A USB device/token that needs to be connected to the computer and sets up a separate connection with the banking server, which is for example used in the ABN AMRO online banking model [5].

Both models show the correct transaction information being the information received by the bank over the out-of-band channel. In addition, the user needs to confirm the transaction with the TAN code received by SMS or on the secondary out-of-band device, through which each fraudulent transaction can be noticed and canceled.

The advantage of two-factor authentication

The following graph gives an idea of the amount of fraud, if only single-factor authentication is used compared to the amount of fraud when two-factor authentication is used. It is presented by the amount of money lost by a bank, considering the number of clients. However, the graph does not cover the amount of fraud considering any model with out-of-band transaction control, because no data was available.

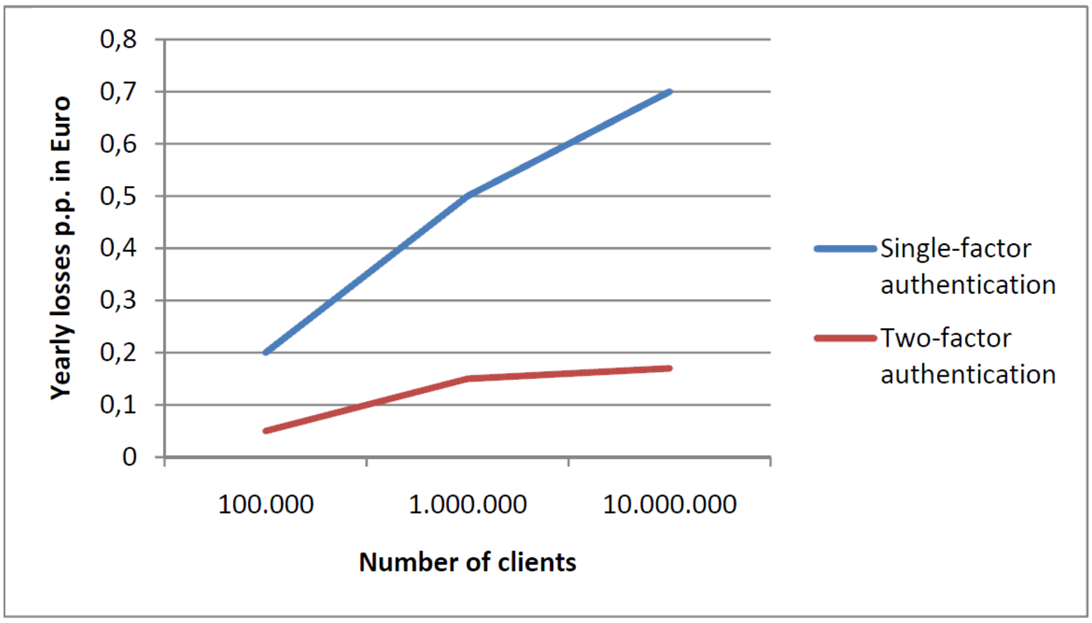

Figure 1: Fraud level overview: One- and two-factor authentication

The fact that both the single-factor authentication and two-factor authentication losses per account increase as a function of the size of the bank can be explained by the fact that a bigger bank is a more attractive target: the attackers need a certain size bank to "break even" on the attack investment, and have econonomies of scale from thereon, so "bigger is better", and fraudsters aim for the big banks.

More tricky is the fact that the single-factor authentication losses increase more or less linearly, while the two-factor authentication losses clearly flatten off at around 1M customers. That is probably best explained by the fact that single-factor authentication credentials can be harvested, stored, and traded, whereas two-factor authentication credentials have a "due date" that is measured in minutes: they have to be (ab)used in near real-time. Since the bottleneck for transaction fraud is typically the cash-out (e.g. via money mules), it brings an advantage for the single-factor authentication phisher to be able to trade in stolen credential batches, in effect outsourcing the cash-out. Hence, single-factor authentication fraud per account scales linearly with the size of the bank: the bigger the bank is, the more attractive it is to target, so the more fraud occurs, not only for the bank as a whole, but even per account. For a two-factor authentication fraud this is not the case: at around 1M customers, the fraudsters run into a cash-out scalability issue, and lose the incentive to chase bigger banks.

3 Multi-layer security

This report has already demonstrated that most of the currently used models have an inadequate protection against the newest generation of malware (MitB). This section will introduce our three-layer security approach and present different useful methods per layer, which can be used to counter (malware) attacks. Some of these methods, however, are based on hampering the malware writers and can not be viewed as a permanent solution. Examples of these temporary solutions are the use of dynamic text and the use of captcha's.

3.1 Layer I: End-user PC

3.1.1 Methods

Hardened Browser:
A web browser which has fewer options and does not allow the use of any external extention or add-ons/Browser Helper Objects (BHO's). This should reduce the threat of Man-in-the-Browser attacks. Such versions are not yet supported by most browser vendors. Though some web browsers, however, such as Google Chrome and Apple Safari do not support (external) add-ons yet and are therefore indirectly protected to be exploited by a Man-in-the-Browser attack. The web browser Internet Explorer (IE) 8, which does support add-ons, has an option to disable the use of its BHO's: "InPrivate browsing". Mozilla Firefox (FF), which also supports add-ons, has a slightly more complicated method for the user to disable its add-ons. In its executable properties, the command option needs to be completed with: "-safe-mode". In safe mode, the add-ons will be disabled and the option is presented to also disable all add-ons in every standard

session. Even though IE and FF can not really be viewed as hardened browsers, it is a step in the right direction which already stops a part of the problem.

Virtual Machine:
VMware has a great definiton of a, fully virtualized, virtual machine:

> A virtual machine is a tightly isolated software container that can run its own operating systems and applications as if it were a physical computer. A virtual machine behaves exactly like a physical computer and contains it own virtual (ie, software-based) CPU, RAM hard disk and network interface card (NIC).
>
> An operating system can't tell the difference between a virtual machine and a physical machine, nor can applications or other computers on a network. Even the virtual machine thinks it is a "real" computer. Nevertheless, a virtual machine is composed entirely of software and contains no hardware components whatsoever. As a result, virtual machines offer a number of distinct advantages over physical hardware[8].

In a first kind of virtualization, a computer boots into a new virtual environment on which multiple virtual machines can run. Examples are XEN and VMware ESX. Other virtualization products like VMware Workstation, VirtualPC and VirtualBox are run from a standard OS and are as such still susceptible to OS malware as they are in fact a normal application. Using technology like Java Web Start it is possible to push the virtual environment to a user, this technology is the basis of our future model 1 which will be explained in section 4.3.

Secure Signed Users:
A user signs or encrypts the data by means of a separate program or device, before sending it to the server through the web browser. Using this method, data sent from the user's computer cannot be manipulated, but OS malware that resides on the user's computer can still change the data.

Delayed Webpage Delivery:
By delaying the loading of the original webpage, the changes that are made by malware might be revealed by changes in the appearance of the page on the client side. Malware can, however, easily be altered to wait until the page is fully loaded by the user before making any changes to the webpage. This is why the method would only temporarily counter a number of known attacks.

Transaction Model:
Its strategy lies in using a lot more dialog between the user and the server in order to make it harder for a Man-in-the-Browser to generate all answers. Current malware already bypasses this by letting the user fill in all of the information and changing only the transaction vital data.

User Education:
Educating users may help in getting them familiar with the existing threats and how these can be countered. This can be done e.g. by presenting guides or by presenting information in the form of commercials. Organisations do not often use this method because many attempts of educating users simply do not succeed. This possibly means that the current way of educating people is not appealing to most of the userbase.

Server Side approaches:
For a secure authentication to the end-user PC, some security methods exist that can be imposed from the server-side, with the goal to circumvent the working of malware at the user-side.

Random or Dynamic Text:
Randomly changed URL, pages, fields and other elements make it harder for malware to duplicate and interact with a webpage. Downside is that users often get a different webpage, which makes it harder for a user to recognize a phishing site from a genuine one.

Pre-Encryption:
The server sends its data encrypted, which means that the data can not be changed before it reaches the user. In this method, the encrypted data is also not decrypted by the web browser, it must be done by a secundary application. As such, this method has the disadvantage that the user needs to learn how to decrypt the data in a secure way. This might be done by a separately installed application which needs the encrypted data as user-input or by using a BHO/add-on, which automatically decrypts the data. The latter has more user convenience, but this is only gained by reducing the security aspect, because it opens the possibility of a Man-in-the-Browser attack.

Captcha's (Completely Automated Public Turing test to tell Computers and Humans Apart):
Captcha's are images with letters and numbers which are generated and presented to the user. The user needs to enter these letters and numbers into the intended field, before being able to proceed to the following webpage. It is particulary useful for denying access to automated scripts (bots), but it is easy to bypass with malware, such as Man-in-the-Browser malware, which only changes vital data.

3.1.2 Analysis

The methods that are being pushed from the server side are of a more temporary nature, which makes them as such less interesting to be implemented in a new model. More promising methods are for example the virtual machines and the hardened browsers. The virtual machines because it provides a secure environment and the hardened browsers because it counters the most common attack of the moment, namely a Man-in-the-Browser attack. Obviously, these methods need to be implemented in combination with other methods to provide a complete model, covering multiple security aspects.

3.2 Layer II: Extra out-of-band authentication

From our point of perception, it is assumed that the end-user PC might be compromised and that this may or may not be (completely) resolved. This is why a second layer is being considered. This layer is used to garantuee an additional safe authentication through the use of a secondary channel. Because this channel is separated from the main communication channel, mostly being a connection originating from the web browser, it can not be compromised likewise.

3.2.1 Methods

- USB-connected SmartCard Reader - These readers have a LCD screen and a secure login method, usually using a pincode. If connected to the USB-port during the transaction, a secundary (secure) channel is set up for transaction control.
- Read-only Media.
- External Authorization Device - This means that an external device is needed to authorize a transaction using e.g. a code that needs to be entered before the transactions in finalized. Short Message Service (SMS) is often used because nowadays nearly everyone owns a mobile phone.
- Transaction Based Applications Using a Second Channel - An application that uses a second connection e.g. SSL/TLS, next to the connection originating from the web browser.
- Interactive Voice Response (IVR) - This method is based on making a telephone call to the customer by a callserver, to authenticate a transaction or make a change in the user's profile.

3.2.2 Analysis

The above mentioned methods have all been implemented in some existing models, however they do not have the same effectiveness against the current attacks. External authentication devices have the advantage that they are not in any way connected to the (potentially) compromised computer. This prevents any OS malware to intervene in the communication between the bank and the external device. It sounds like the perfect solution, however its cost is such a major drawback that many banks are not convinced to use it.

The IVR method is great as a out-of-band transaction control, but has multiple disadvantages. For example, the user is obliged to have his phone with him, whenever he needs to make a transaction. Another problem is that the possibility to change the phone number must be well thought out. If it would be possible to change the phone number on the website, it opens a window of opportunity for the attacker. If the phone number can only be changed by a confirmation call on your previous phone number, it would be problematic when the phone is lost or stolen. Furthermore, a phone call would always be more expensive as for example SMS. On the other hand, because IVR supports two-way comunication, it could be perfectly used for a small percentage of transactions which have a supposedly high fraudulent score (See Layer III: Back-office monitoring, section 3.3). This method could then summarize the transaction and ask for further authentication in the form of a part of the recipient account, a pincode, a short-time or one-time password or any other security question which can be numerically answered.

Read-only media like USB devices are often used in combination with transaction based applications which use a second channel, such as a SSL connection instead of the web browser to execute transactions. The fact that it prevents Man-in-the-Browser attacks makes it an interesting method, which must be considered for future models.

The USB-connected smartcard readers have the most potential because they implement a challenge response system, which prevents keyloggers from getting vital information, and if connected to the USB-port they implement a secure secondary channel, which protects against possible Man-in-the-Browser malware. This is, however, considered to be very expensive.

3.3 Layer III: Back-office monitoring

Back-office monitoring is detecting fraud from the side of the bank, which does not need any interaction from the client. Momentarily, there are a number of products such as RuleBurst[16][17] and Verisign Identity Protection Fraud Detection Service[15], which often include a rule-based fraud detection system and self-learning modules, that can adapt to certain user profiles and the behavior of new forms of malware. Back-office monitoring might, however, have problems with detecting certain attacks, such as Man-in-the-Browser attacks, which only changes a certain part of the user input. Banks might also have certain objections to the implementation of back-office monitoring, because most monitoring packages currently work against money laundering or for the purpose of fraud-detection. It would be a costly investment for any bank if two monitoring packages must work parallel to eachother.

Fraudulent attacks are mostly directed to multiple banks at the same time. This means that the advantage of back-office monitoring could be increased, if banks would share their collected data directly or by means of a third party. As a consequence, new attacks can be discovered more easily and as such, can be countered more quickly.

A number of rules or methods which are used within back-office monitoring are:

- Session checking (using e.g. GeoIP to locate a session)
  - Session location - This is useful, because a person can not travel any distance within a certain time.
  - Number of simultaneous sessions - To disallow multiple transactions from one client at the same time and to detect brute forcing attempts.
  - Number of different sessions from one IP address - To check if one IP address makes a predefined number of sessions using credentials from different clients. This would have to be checked with a "smart" algorithm, to make sure that the IP address does not belong to a proxy server, covering multiple clients.
- Cookies - Detect if multiple clients connect from one computer within a specified time frame.
- User profiling - Usually this works like a scoring card where every deviation from "normal" behaviour scores a number of points, upon which a session can be cancelled if a certain limit is reached.
  - Modem type - By using a trace route detect the type of modem the client connects with.
  - Connection speed - Internet connection speed at the location of the client.
  - Location - Client location often has a pattern. For example in the morning they connect at work and in the evening they connect at home.
  - Browser type - Changing browser type within a specified time frame or session.
  - ...

Even though back-office monitoring is hardly used for anti-fraud purposes nowadays, we strongly emphasize its future importance for the opportunities which it creates. These are amongst others the possibility to estimate the amount of fraud with more accuracy than before and the possibility to counter some of this detected fraud before the fraudulent transaction could have been completed.

4 Next generation models

As we have stated before, most of the current models are not sufficiently secure against the latest threats. This is beside the models from ING and ABN AMRO which we deem to be secure and able to counter this new threat. However, this does not mean that these are the best possible models to be used and for this reason, this section covers some newly established models. We have taken two approaches to establish these models. In our first approach, we have used existing models and combined these with some of the methods presented in section 3. The idea was to offer a more complete protection, by chosing models which were already sufficiently secure against the older known threats and combined it with methods which protect against the newly risen threat of Man-in-the-Browser malware. For our second approach, we have used a combination of multiple

methods, which are situated in the different layers and were presented in section 3.
All added methods, which are used in one of the following models, are part of the first two out of three discussed layers:

- Layer I: End-user PC
- Layer II: Extra out-of-band authentication

This comes forth from the fact that we have searched for a level of security, which gives the best possible protection, independent from a necessary combination of all three layers. This has brought forth that whilst back-office monitoring provides many advantages, it is best used as an extra on top of one of these models, for the reasons which have been provided in the previous section. To make it possible to compare our own models afterwards with the models from ING and ABN AMRO, we have first included a description of these existing models.

4.1 ING model

The ING model is one of the few existing models which is able to counter Man-in-the-Browser malware. The first step, logging in, still happens with a challenge-response token, but it is the transaction which has an extra security measure: Out-of-band control messages (SMS). If a transaction is made, a SMS message will be send with all the transaction information and a TAN code for authorization. If the information in the SMS is correct, the TAN code can be entered at the ING website to authorize the transaction. One of the possible disadvantages of this model is that it is dependent on the network of the secondary channel. Would the network be congested, such as frequently happens during New Year's Eve, it would not be possible to do any online banking during that period.

4.2 ABN AMRO model

The ABN AMRO is our second example of an existing model which is able to counter Man-in-the-Browser malware and this with the use of dual-channel authentication. The e.dentifier2 has two modes of operation: the unconnected and connected mode, of which the second mode is definitely the most secure.

Unconnected mode:
Online banking is possible from every connected computer when using the unconnected mode. For logging in, first the account number and the card number must be entered to authenticate the user, after which the bank card must be inserted in the e.dentifier2, and the PIN code to authenticate. The e.dentifier2 will then give a response which must be entered on the ABN AMRO website to finalize the logging procedure. For each transaction, a new challenge-response will need to be executed for a proper authentication. This mode can be compared with a normal two-factor authentication model, without the use of out-of-band authentication and is as such not secure against Man-in-the-Browser malware.

Connected mode:
In the connected mode, the e.dentifier2 is connected with a USB cable to the USB port of the computer. To be able to work, it needs a one time installation of software, but it also allows the users to skip the response step and the user authentication in the beginning (account number and card number). All what is needed to do, is go to the ABN AMRO online banking site, connect the device to the computer, insert your bank card and enter your PIN code. After this, the e.dentifier2 asks permission for every step using the secure secondary channel. This is for logging on and for making a transaction, for which the transaction information is shown on the e.dentifier2 screen. Only this mode protects against Man-in-the-Browser malware.

4.3 Model 1: Thin server-side virtual machines

The thin server-side virtual machines model is based on the assumption that a certain percentage of the user-base is connecting to their online banking account by the means of a compromised end-user PC. Instead of trying to resolve it or trying to circomvent it with out-of-band authentication, we chose to bring the banking application back to a more secure and controled environment: the server-side. The server-side is able to provide a virtual environment which increases the level of security, because this environment can be completely stripped of everything which is not useful for the application. The user will then, upon connecting to the banking site, only need to load a small application which makes a secure connection to the virtual machine. For example, could this be done through the use of a Java applet which is pushed with Java Web Start. This secure environment does two things:

- It makes Man-in-the-Middle attacks more difficult, because the attacker has to create an application and data from that application has to be interpreted/transferred to the genuine bank application and visa versa.
- It makes Man-in-the-Browser attacks impossible, because the application is used instead of the web browser.

Another necessity is to use a small challenge-response token. This counters the possibility of an attacker stealing the credentials with a simple keylogger, which would enable him to log in to the application from any specific location. Aside from this, another advantage is that if any problem would ever turn up it can more easily be countered because the control has been brought back to one point: the server-side. This model has, however, no special measures implemented against pharming and phishing attacks. From our point of view, users are nowadays more educated than before on the danger of phishing sites which makes them more suspicious of malicious e-mails. Another point, which speaks in favor of this model, is the difficulty to spoof the banking site with its application, as it requires a much greater load than is needed for spoofing a static website.

4.4 Model 2: Hardened browser against Man-in-the-Browser attacks

The hardened browser against Man-in-the-Browser attacks model is aimed as an addition to existing two-factor authentication models which are commonly used by most banks. These current models were sufficiently secure until the rise of the Man-in-the-Browser malware, which could not be countered by using just two-factor authentication. This model has the current model of those banks as a starting point, but added with our "hardened browser" method to counter the Man-in-the-Browser malware. As stated, a truly hardened browser does not exist yet, but the mentioned possibilies should be sufficient to counter the threat. That is what makes this model very practical to implement for banks, because all they need to do is educate their users to do their online banking with:

1. A browser which does not support add-ons/BHO's: e.g. Google Chrome or Apple Safari.
2. "InPrivate browsing" in Internet Explorer 8, with the following option enabled: "Disable toolbars and extensions when InPrivate Browsing starts".
3. "-safe-mode" added to the command option in the executable properties, if using Firefox.

A side-effect of this model is that it makes banks once again dependent on the safety of web browsers and as such of their manufacturers.

4.5 Model 3: Trusteer against Man-in-the-Browser attacks

The Trusteer against Man-in-the-Browser attacks model has the same starting point as model 2 but instead of a hardened browser, an application is installed. This application does the following:

- Prevents malicious/unknown add-ons of a browser to execute (to prevent Man-in-the-Browser)
- Prevents users from entering their credentials, which they use on a protected website, on non-protected websites.

An example of such an application is Trusteer Rapport[6], which divides websites into three groups: protected, grey area and unknown websites. A key characteristic is that credentials which are used on a protected website can not be used at another website, and as such ensures that banking credentials are not reused at for example forum websites or spoofed websites.

For logging in, this model uses a username/password and a simple token that generates a one-time/short-time password. The token is used to prevent keyloggers from capturing (useful) credentials. The drawbacks of this model are that an application needs to be installed, which is possibly only supported by a number of operating systems, and that user education is needed to ensure a proper installation and use of Trusteer. Another disadvantage is that when installing Trusteer using a non-admin account Trusteer is unable to install all security features. Whilst online banking would still be possible without Trusteer, the level of security would dramatically decrease.

4.6 Model 4: USB-application

The USB-application model is based on an application, which is placed on a USB device and where the purpose of the application is to bring the transaction to a secure channel. The application uses an SSL connection to connect to the bank and all data that needs to be sent through the secure channel will get encrypted with a key. The key will be asymmetric, because of the fact that this type of metric has scaling advantages and these keys can be activated and revoked more easily.

For logging on, this model uses a username/password and a token to generate a one-time/short-time password. As in the previous model, this prevents the use of keyloggers from stealing any (useful) credentials.
One of the drawbacks of this model is that OS malware is still a problem, because it does not use any form of out-of-band authentication and the application is run on top of the computers OS. Another problem lies in the fact that the USB port is not always accessible for a user, like in a internetcafe or a business environment. Users also need to carry both a USB and a token device for online banking, which makes it less practical.

4.7 Model 5: Processor-based USB-application

The processor-based USB-application model is based on the IBM ZTIC (Zone Trusted Information Channel)[7]. This product consists of a USB device with an internal processor and an LCD. The procedure to make a transaction goes as follows:

- Insert the USB device
- Connect to the banking website
- Log in using username/password and a response code sent to the USB's LCD by means of a secundary secure connection
- Fill in the transaction information
- Check the transaction information by using the USB's LCD screen
- Click on the OK button from the USB device to complete the transaction or on the CANCEL button to cancel the transaction
- Eject the USB device

After inserting the USB device, it starts a "pass-through" proxy which is configured to connect with a pre-configured (banking) website. After starting the proxy, the user opens a web browser to establish a connection with the (banking) website. Because of the response code, which is used during the login, credentials received using a keylogger do not enable the criminal to login. The secure second channel makes it possible to check the transaction information before completing it, which prevents a Man-in-the-Middle or Man-in-the-Browser to manipulate a transaction.

5 Model analysis

5.1 The most balanced model

To find the most balanced model, we have compared our "new generation models" and the two-factor authentication models of ING and ABN AMRO by means of three factors: cost, user convenience and security. For this comparison, our corporate research partner has provided us with the necessary information to make an estimation of the costs of each of the models. Please note that the ING and ABN AMRO costs are not actual costs, but modelled costs. We modelled theses costs so we could make a like-for-like comparison between these "real life" models and our proposed models. Detailed information on our methods of calculation can be found at appendix B.1. In order to give a clear overview and comparison we have placed the results in a graph, as can be seen in figure 2.

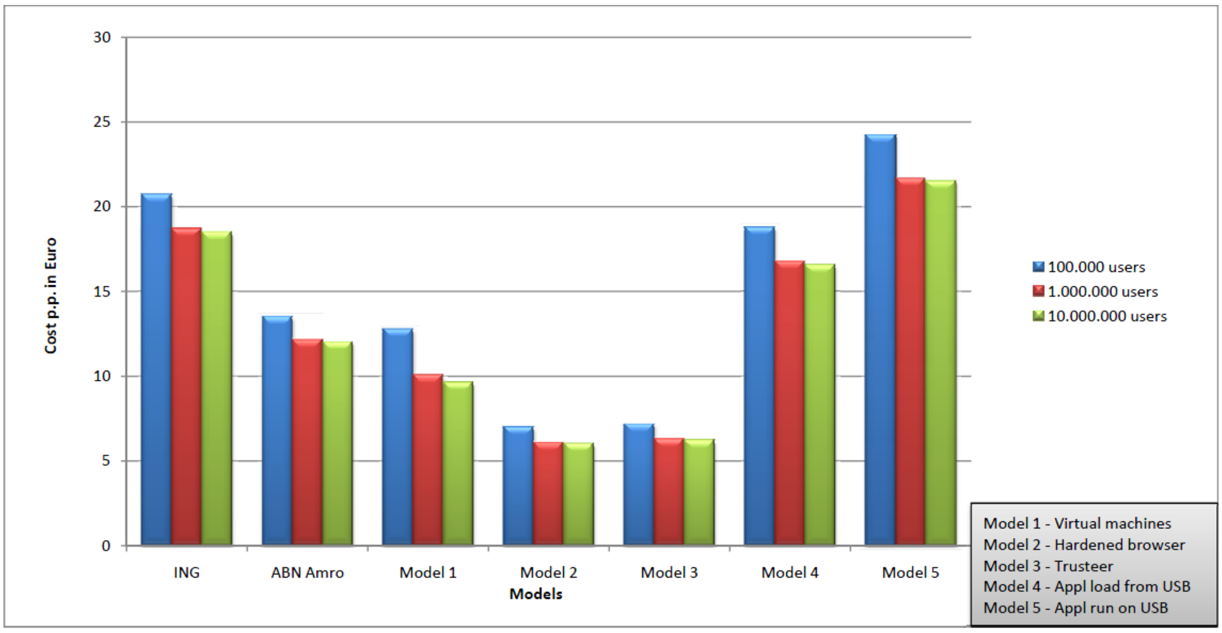

Figure 2: Estimated cost overview

From a cost only perspective, models 2 and 3 clearly have the upperhand, whilst models 4 and 5 and the ING model peak with the highest cost per person. These higher costs are mainly caused in models 4 and 5 by the use of a rather expensive device, being the combination of a token and a USB device in model 4 and a USB device with an internal processor in model 5. The ING model's higher costs are due specifically to the use of SMS messages as an authentication and control mechanism.

In order to be able to compare the level of user convenience and security, we have created a number of questions on which we based the results. Detailed information about these questions, how we divided the points and how we came to our results, can be found in appendix B.2). Using this information, a graph has been made that shows the scores on a total of 10 points. The maximum score can, however, never be reached by any model, because it is a summation of their advantages and disadvantages given a certain scale. As such, the scores can not be regarded as values, because it is rather a comparison between the models of which one scores the best on security and user convenience.

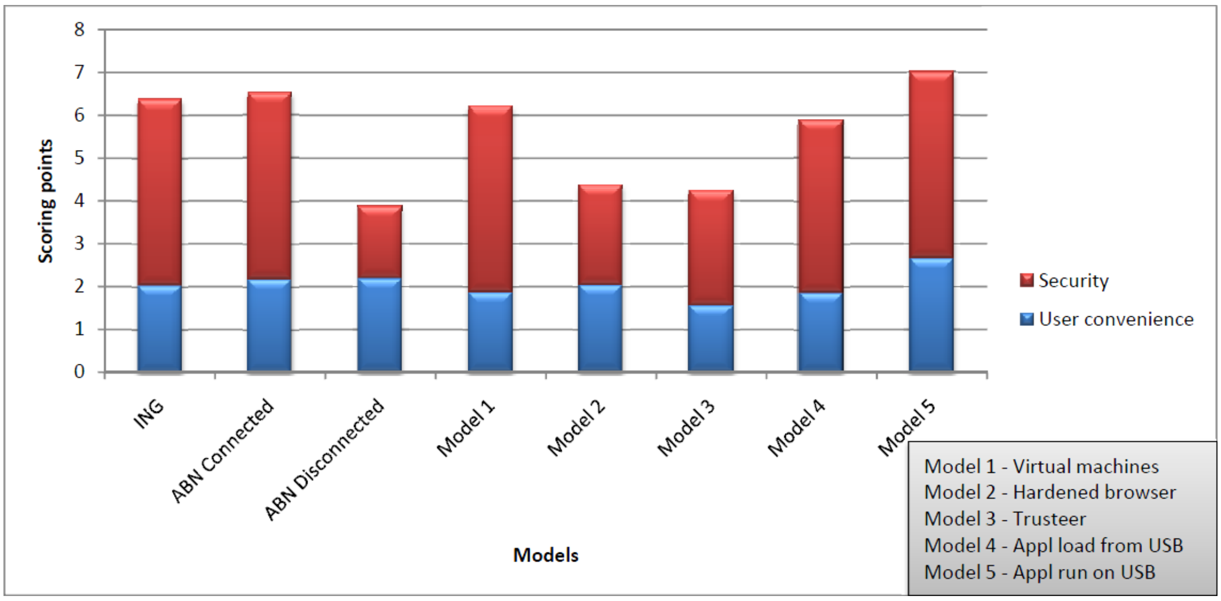

Figure 3: Convenience & Security overview

After analysing both graphs and their data, we consider the first model (Thin server-side virtual machines) to be the most balanced model. The fifth model may have the highest ranking in the user convenience/security ratio, but the extreme cost per user makes it unusable. The costs of the second and third model are considerably lower, even in comparison with the first model, but the level of security has been the decisive factor for us. The main reason for our decision is that both models (2 and 3) are mostly dependent on the proper use by their users, considering the "hardened browser" or "Trusteer" method. These methods also make the security of the banking transactions too dependent on the security of the web browser or from the security of the Trusteer product. Altogether, we still advise any bank, which momentarily uses a standard two-factor transaction and user authentication model, to implement one of these models as a temporary quick fix. This gives their online banking model protection against the new malware threat of Man-in-the-Browser and gives them the necessary time to switch to a more foolproof model, such as our "thin server-side virtual machines" model.

Comparison with the current models

Whilst the online banking model of ABN AMRO is not one of the more expensive models, it had lost our interest because of its dependence on the user awareness. Whilst the

disconnected mode of ABN AMRO offers the most user convience, it works by lowering the level of security through sacrificing the second communication channel. Our fear is that the unaware user will prefer the option which brings more user convenience, because it needs fewer steps to perform a transaction and hence makes it faster. The more secure option is also possible only if the device can be connected to a USB-port, which may make the secure option impossible in certain places (e.g. workplace or internetcafe).
The online banking model of ING still has an acceptable level of user convenience whilst keeping its level of security rather high. This level is, however, partially dependent on the level of user awareness, as a user needs to check if the transaction information, which he receives in an SMS, is correct. Only then, should he enter the received TAN code at the website. We believe that this control process is rather practical for most users and is therefore sufficiently reliable. The main disadvantage of the online banking model of ING is the cost factor, which is caused by the use of SMS's as an external transaction authentication and control mechanism. It is an excellent tool for the transaction authentication and control, but banks who use this model should consider whether the high costs are justified.

5.2 Future malware threats

Criminals have always had a way of breaking new security models, which makes it seem that there can never be a foolproof model. This is why this section is dedicated to discuss a possible attack that could be used in the future to exploit our "most balanced model".

Man-in-the-Middle
A phishing or pharming attack could be used to mimic the genuine website, with the purpose of fooling the client in downloading the criminal's application instead of the genuine banking application. The criminal could set up his/her own connection to the bank and relay all challenge-response information to the user using his/her own application and use the information the client enters to make a transaction of his/her own. Such an attack has the following requirements to be able to succeed:

- The application needs to have the "look and feel" of the genuine application.
- The spoofed application must:
  - Be run from a computer with a fast connection.
  - Have both a connection to the user as well as to the genuine banking application.
  - Relay information within a certain time span:
    - User credentials, to falsly log in and make a transaction.
    - User information, to trick the client into believing to be viewing the genuine virtual machine.

These requirements make it very hard to sufficiently scale this kind of attack, because of the high load.

OS malware
Where the current websites can be manipulated, with MitB malware, through the use of HTML injection and sending fake data to the website, it gets a lot harder when trying to manipulate a thin server-side virtual machine. In this case the malware needs to go one level up to the level of the OS. Keystrokes could still be intercepted and changed with other keys to change an account number, an amount of money, etc. The hard part is, however, that the application which is residing on the server-side displays itself at the user-side by only sending over a screen which represents its GUI. With other words, this means that the malware must be able to alter the receiving bitmaps on the fly, to be able to hide the fact that some numbers or strings might be altered. Whilst this is a ”possible” attack, we deem it very hard to successfully launch it.

One of the main issues with the previously introduced attacks is that they have to be uniquely written for each bank which uses the thin server-side virtual machine's model, whilst on the other hand, current malware is able to exploit multiple banks at the same time. If we hold this fact next to figure 1 from section 2.2, which indicates that banks with more clients are more susceptible to fraud, this indicates indirectly that this model makes a bank less attractive for an attacker, because it limits their scale of attack.

6 Conclusion

During this research project, we have stated that two-factor authentication methods are not longer sufficient as a stand-alone security measure. This is due by the rise of Man-in-the-Browser malware, which has triggered the need for finding new security implementations. Whilst current online banking models from ING and ABN AMRO are well suited against this new malware threat from a security point of view, we have taken a three-layered approach to find the most balanced model considering cost, user convenience and security.

With regards to the first layer, the end-user PC, we have taken a special interest in two methods. The virtual machine method, which assumes a clean and controllable environment to be used for online banking and the hardened browser method, which makes the currently most vulnerable part of the computer, the web browser, more secure.

The second layer is a consideration for extra out-of-band authentication, in case the end-user PC cannot be fully protected. Two methods seemed to be able to increase the level of security considerably, but are rather expensive to implement. The USB-connected smartcard readers, because the device is so expensive and the external authentication devices, because it brings extra costs to each and every transaction (e.g. SMS). Another less expensive but interesting method was the read-only media like USB devices, which make it possible to load an untampered application from any PC with USB access.

The third layer, back-office monitoring, appeared to have many practical purposes which would make it possible to estimate the amount of fraud with more accuracy than before and to counter some of this detected fraud, before the fraudulent process could have been completed. However, a bank would have to acquire a secondary monitoring package to run next to their money laundering monitoring package. This is, however, quite expensive and as long as there are no regulations, banks will not be hurried into implementing a secondary monitoring package. This is why we have especially focussed on the first two layers to create our own models. Although we believe that back-office monitoring is still a very interesting addition to any banking model. Our rules and methods, which we have summarized in section 3.3, might be a good recommendation on what to monitor, if back-office monitoring would eventually be implemented.

With the previous methods and already existing models, we have created six models of our own and compared them afterwards with two existing models from ING and ABN AMRO. Of all of these models, our preference went to the "thin server-side virtual machines" model. Its set-up of moving the application or browser away from the user's side and to the server side, we believe it will greatly increase the level of security. First of all, the server environment is much easier to control and can be restricted to be used solely for the meant application. Secondly, its cost per user compared to the models with approximately the same level of security and user convenience is considerably lower.

We realise, we must emphasize that our solution is not the one and only one around. As much depends on the currently implemented models at a bank and their current level of security. Banks who have most of their online banking security in order, would have a level of switching costs that is too high to switch to our "thin server-side virtual machines" model. If they were using a proper two-factor authentication, they might however prosper with our second model, which only adds the use of a hardened browser at their client sides. This would mean low switching costs and still an added safety mechanism against the latest (Man-in-the-Browser) malware.

Finally, we have taken a look at what kind of malware attacks could possibly be used against our most balanced model, if it would be implemented in the future. Whilst we have come up with two possible attacks, a Man-in-the-Middle attack and a OS malware attack, we believe that they are hard to implement and even harder to implement on a large scale. Furthermore, we indicated that these attacks are not possible on the same scale as for instance Man-in-the-Browser attacks, which can be used for many different banks at the same time. This is why we concluded that this model makes a bank less attractive for an attacker, because it limits their scale of attack. We also see a future for back-office monitoring to limit these kinds of attacks even further.

References

[1] Authentication in an Internet Banking Environment, Federal Financial Institutions Examination Council, FIL-103-2005, October 12, 2005.
http://www.ffiec.gov/pdf/authentication_guidance.pdf

[2] Enhanced Login Security - Online Banking, Northwestern Bank
https://www.nwbank.com/IDTheft/?fuseaction=article&articleid=9556

[3] Online Banking Enrollment Help and FAQs, Pacificcrest Bank
https://olb.pacificcrestbank.com/mfaHelp.html

[4] ING online banking
http://www.ing.nl/particulier/internetbankieren/index.aspx

[5] ABN AMRO online banking solution - edentifier2 user manual
https://www.abnamro.nl/en/images/Generiek/PDFs/Overig/edentifier2_usermanual_english.pdf

[6] Trusteer Rapport
http://www.trusteer.com/solution

[7] IBM Zone Trusted Information Channel
http://www.zurich.ibm.com/ztic/

[8] Definition of a virtual machine by VMware
http://www.vmware.com/technology/virtual-machine.html

[9] An Empirical Analysis of the Current State of Phishing Attack and Defence - Tyler Moore and Richard Clayton - May 11, 2007

[10] The Phishing Guide (Part 1 & 2) Understanding and Preventing Phishing Attacks - Gunter Ollmann - last edit June 07 2009
http://www.technicalinfo.net/papers/Phishing.html

[11] Technical Trends in Phishing Attacks - Jason Milletary -US-CERT

[12] Phishing Scams: Understanding the latest trends - FraudWatch International - June 2004
www.fraudwatchinternational.com

[13] The Pharming Guide (part 1 & 2) Understanding & Preventing DNS-related Attacks by Phishers - Gunter Ollmann - last edit June 07 2009

[14] Drive-By Pharming - Sid Stamm, Zulfikar Ramzan and Markus Jakobsson - December 13 2006

[15] VeriSign Identity Protection (VIP) Fraud Detection Service
www.verisign.com.hk/authentication/guide/VIP_FDS_whitepaper.pdf

[16] RuleBurst - Austalian based Business Rule Management System [dutch link]
http://www.sap.com/netherlands/company/events/overheidspraktijk/pdf/Ruleburst.pdf

[17] Oracle acquires Ruleburst Holdings Limited
http://www.oracle.com/haley/index.html

[18] Point and click Gmail hacking at Black Hat
http://www.tgdaily.com/content/view/33207/108/

A Known attacks

Known attacks often include the term ”identity theft”. This term is used to point out that the attacks use intercepted information to the attackers' advantage. This usually means that the information is used to commit fraud. Attacks that can be used to commit identity theft are for example:

A.1 Man-in-the-Middle

This type of attack consists of a third person joining a conversation by intercepting all information, from both Alice and Bob, in such a way that both victims think they are speaking to each other. Because of this, Trudy - the attacker - is the first who gets all information sent to Alice or Bob and can manipulate this information to her advantage before it reaches the intended receiver. An example which is in line with our research, is that Trudy could manipulate the information which Alice sends to Bob and where Bob is the bank. If a Man-in-the-Middle attack would occure during a bank transaction, Trudy could change the receiving accountnumber and the amount of money which is transferred, whilst making Bob believe that his original transaction has occured without any problem.

A.2 Man-in-the-Browser

Man-in-the-Browser attacks are very simular to Man-in-the-Middle attacks, but the third party resides in the web browser. The browser is compromised in such a way that it displays the user input to the user, but manipulates this input and sends the manipulated data to the website. This kind of malware makes use browser functionalities which allows to install extentions to the web browser. Currently, there are two web browsers which have this functionality:

- Internet Explorer, with its Browser Helper Objects (BHO's)
- Firefox, where they are called Add-ons

What makes this attack especially dangerous is that it is very easy to implement this attack and furthermore, there are many easy-to-use toolkits available which make it even less hard to create Man-in-the-Browser malware.

A.3 Shoulder Surfing

Shoulder surfing is used in all kinds of attacks. One of the best known examples is when people withdraw money at an ATM. Here, the card is copied by a ”scimming” device, but it is useless to a criminal if he does not know the card's pincode. In order to obtain it, he will use the shoulder surfing technique, by literally looking over a person's shoulder or by using some kind of camera to retrieve the pincode. With the copied card in his possession and the knowledge of the pincode, the criminal has all the information he needs to hack the account. This type of attack is, however, also implemented in the world of computers. Instead of literally looking over the shoulder, malware is used which registers every key a person uses. Current key loggers even have to possibility to only register the entered keys, when a certain website of interest, such as a certain banking site, is visited.

A.4 Cross-Site Scripting

There are three basic kinds of cross-site scripting. The first consists of a client-script, which is triggered through some kind of user input. This could for example be some information which is situated in a URL. If the (URL) information is not checked upon execution, the script will be triggered and may generate an additional part of a website to retrieve information from a user (e.g. accountnumber or pincode). This information is then stored, by uploading it to the attacker's computer or to a database. In the second variant, a script is hidden within some user input and send to a webserver. Again, only if the server does not check this information, the script will be loaded in addition upon generation of the webpage. In the third variant, the script is inserted into the database of the destination server itself. This can happen if the user transmits a textfield, with an inserted script into the text. If it is not checked before allowing it into the database, it will be executed on the page of every (potential) visitor who views the website thereafter.

A.5 Phishing

A phishing attack is used to lure someone, with an e-mail or with instant messages, to a fake (banking) website, which is mostly a duplicate of the genuine website and has the intention of tricking the user's login and other vital information out of the unsuspecting user. Current phishing attacks do not only retrieve the information, but often they immediately use the login information to set up a session with the genuine (banking) website. Then, step by step, more information is tricked from the user, with which fraudulent actions can be performed, such as making a fraudulent transaction. Currently, this kind of attack makes it possible to bypass most of the used authentication methods. [9][10][11][12]

A.6 Pharming

A pharming attack has the same purpose of a phishing attack, luring an unsuspecting user to a fake (banking) website, but it is accomplished in a different way. A pharming attack poisons a DNS-server or a local name resolving method, such as the *"\etc\hosts"* file, by changing the IP-address where a specific domain name resolves to. Then, if a user wants to visit this website, the valid URL will be resolved to the altered IP-address, on which the user will be refered to the website of the attacker. [13][14]

A.7 Session (Cookie) Hijacking

Session hijacking is done by stealing a session ID, which is mostly located in a cookie, from a client computer. This is then used by the attacker to duplicate that session.

An example of this attack has been given by Robert Graham at a Black Hat security convention [18], where he demonstrated how e-mail accounts can be hacked with the use of session hijacking.

A.8 Operating System Malware

OS malware consists of a malicious application or code that among other things reads directly from the Random Access Memory (RAM). This means that even encrypted messages can be read, because the processor has to decrypt it at a certain point. Another element of this kind of malware consists of intercepting system calls to the OS and by doing so, it can replace every system call by its own. A combination of these two elements can have as a consequence that an encrypted message can be resolved to a decrypted message, which on its turn can become manipulated. This category of malware contains among other things:

- OS based keyloggers
- Network sniffers
- Rootkits (They hide themselves and the data which they have collected in the OS)

B Calculating the most balanced model

All models have been given a score on cost, user convenience and security to be able to compare them. In section B.1, the calculation method and results are presented considering the cost of each of the used authentication methods and models. Section B.2 then presents the calculation method and results considering the user convenience and security for each of the discussed models.

B.1 Cost

B.1.1 Cost per authentication method

Our corporate research partner has provided us with the necessary information which was needed to determine the cost per authentication method. For every method, the following formula has been used:

```
A = Initial user provisioning cost / 5 (for an equal division per year)
B = Additional user reprovisioning cost per year
C = Datacenter base cost per year
D = Datacenter marginal cost per 100.000 users per year
E = Extra costs per transaction
nUsers = Number of users
xTransactions = Average number of yearly transaction per user (= 136)

Method cost = (A + B) * nUsers + C + (D * (nUsers / 100.000)) [+ E * (nUsers * xTransactions)]
```

Password:

$$(1 + 2,5) * nUsers + 25.000 + 1000 * (nUsers/100.000)$$

When using this formula the cost with X number of clients can be calculated. This results in the following:

100.000 users: 376.000

1.000.000 users: 3.535.000

10.000.000 users: 35.125.000

Token:

$$(3 + 3) * nUsers + 100.000 + 2000 * (nUsers/100.000)$$

When using this formula the cost with X number of clients can be calculated. This results in the following:

100.000 users: 702.000

1.000.000 users: 6.120.000

10.000.000 users: 60.300.000

LCD token:

$$(6 + 6) * nUsers + 150.000 + 3000 * (nUsers/100.000)$$

When using this formula the cost with X number of clients can be calculated. This results in the following:

100.000 users: 1.353.000

1.000.000 users: 12.180.000

10.000.000 users: 120.450.000

SMS:

$$(1+2,5)*nUsers+100.000+2000*(nUsers/100.000)+0,04*(nUsers*xTransactions)$$

When using this formula the cost with X number of clients can be calculated. This results in the following:

100.000 users: 452.000 + 544.000 = 996.000

1.000.000 users: 3.620.000 + 5.440.000 = 9.060.000

10.000.000 users: 35.300.000 + 54.400.000 = 89.700.000

USB/Application:

$$(2 + 5) * nUsers + 100.000 + 2000 * (nUsers/100.000)$$

When using this formula the cost with X number of clients can be calculated. This results in the following:

100.000 users: 802.000

1.000.000 users: 7.120.000

10.000.000 users: 70.300.000

IBM: display + Processor:

$$(9 + 9) * nUsers + 150.000 + 3000 * (nUsers/100.000)$$

When using this formula the cost with X number of clients can be calculated. This results in the following:

100.000 users: 2.053.000

1.000.000 users: 18.180.000

10.000.000 users: 180.450.000

VM server cost:

```
A = Datacenter base cost per year
B = Yearly marginal costs per extra server
nServers = Number of needed servers. One server is taken per 100 concurrently sessions

Method cost = A + (B * nServers)
```

$$100.000 + 10.000 * nServers$$

When using this formula the cost with X number of sessions can be calculated. This results in the following:

100.000 users (1000 sessies): 200.000

1.000.000 users (3000 sessies): 400.000

10.000.000 users (9000 sessies): 1.000.000

B.1.2 Cost per model

To calculate the cost per model, we have added the cost of each of the authentication methods which are used in that particular model.

ING model: username/password + token + sms

100.000 users: 376.000 + 702.000 + 996.000 = 2.074.000 (20,74 per client)

1.000.000 users: 3.535.000 + 6.120.000 + 9.060.000 = 18.715.000 (18,72 per client)

10.000.000 users: 35.125.000 + 60.300.000 + 89.700.000 = 185.125.000 (18,51 per client)

ABN AMRO model

100.000 users: 1.353.000 (13,53)

1.000.000 users: 12.180.000 (12,18)

10.000.000 users: 120.450.000 (12,05)

Model 1: Thin server-side virtual machine: username/password + challenge/response token + extra server cost

100.000 users: 376.000 + 702.000 + 200.000 = 1278000 (12,78 per client)

1.000.000 users: 3.535.000 + 6.120.000 + 400.000 = 10055000 (10,06 per client)

10.000.000 users: 35.125.000 + 60.300.000 + 1.000.000 = 96425000 (9,64 per client)

Model 2: Token with lcd (challenge/response) + inprivate browsing

100.000 users: 702.000 (7,02 per client)

1.000.000 users: 6.120.000 (6,12 per client)

10.000.000 users: 60.300.000 (6,03 per client)

Model 3: Trusteer + token with lcd (challenge/response)

100.000 users: 20.000 + 702.000 = 722.000 (7,22 per client)

1.000.000 users: 200.000 + 6.120.000 = 6.320.000 (6,32 per client)

10.000.000 users: 2.000.000 + 60.300.000 = 62.300.000 (6,23 per client)

Model 4: Username/password + token + usb

100.000 users: 376.000 + 702.000 + 802.000 = 1.880.000 (18,80)

1.000.000 users: 3.535.000 + 6.120.000 + 7.120.000 = 16.775.000 (16,78)

10.000.000 users: 35.125.000 + 60.300.000 + 70.300.000 = 165.725.000 (16,57)

Model 5: Username/password + IBM usb (processor+display)

100.000 users: 376.000 + 2.053.000 = 2.429.000 (24,29)

1.000.000 users: 3.535.000 + 18.180.000 = 21.715.000 (21,72)

10.000.000 users: 35.125.000 + 180.450.000 = 215.575.000 (21,56)

B.2 User convenience & Security

The models have been compared using a number of questions, which are presented in section B.2.1. Section B.2.2 then presents how these questions are graded and section B.2.3 finally presents the results of each of the models. These results are also visualized in figure 3, which can be found at section 5.

B.2.1 Method of calculation

To be able to compare the models as objectively as possible, the following questions have been used to make an estimation about the level of user convenience and security.

User convenience:

- The number of steps / operations (for the customer):
- The time needed to login and make a transaction:
- Amount of data to remember:
- The number of physical items to keep (physical hardware token and/or token reader):
- The number of different systems used simultaniously:
- The familiarity with the solutions (by other sites / banks) [low, medium, high]:
- Do customers need training / education in using it:
- Calls into the Call Centre (user does not understand what to do / user has technical difficulties) [low, medium, high]:
- Learning Curve [low,medium,high]:
- Different behaviour compared to "normal" Internet browsing:
- Necessary software installations on customer PC:
- Operating System / browser dependency:
- Device dependancy:
- Is the solution "perceived" to be secure:
- Ability to adapt the software/solution to the "look and feel" of the bank :

Security:

- The number of attacks it does not counter:
- Degree of difficulty to perform possible attacks:
- User skill-level/awareness dependence [0-3]:
- Maturity [0-4]:

B.2.2 Questions scoring card

Information found in this section introduces the way the questions have been graded. A lot of questions are in the form of an answer and not a number, because of this reason the answers are transformed to a number with which the user convenience and security of a specific model can be calculated in relation to the other models.

Calculation method user convenience:

- Questions where the answer is a number are used as they are
- Procedure time in seconds 0-5 score where 0 is best
- Where low is a better answer [low,medium,high] = [0,1,2]
- Where high is a better answer [low,medium,high] = [2,1,0]
- Level [easy,medium,hard,very hard] = [0,1,2,3,4]
- Yes/no questions. A negative answer = 1 and a positive answer = 0

Calculation method security:

- Ability to counter attacks 0-5 where 0 is the best
- Level [easy,medium,hard,very hard] = [3,2,1,0]
- User skill-level/awareness dependence where 0 is the best
- Maturity [0,1,2,3,4] where 0 is best (points reversed from questions, changed to lower is best)

Maximum points:
Some questions do not have a fixed number of points and for this reason, the highest number entered at that particular question has been taken as the maximum points that can be received.

User convenience: 8,5,1,2,3,2,1,2,2,1,1,1,1,1,1 [32]

Security: 5,3,3,4 [15]

Formula:
User convenience and security points are being placed in the graph according to the following formula:

$$5 - ((modelpoints/maximumpoints) * 5)$$

Because of this formula the maximum amount of points to receive in total is 10.

B.2.3 Calculated results

This section presents all the grades on user convenience and security which have been given to each model, using the questions found at section B.2.1 and using a scoring mechanism that can be found at section B.2.2.

Model ING:
User convenience:

- The number of steps / operations (for the customer): login:4, transaction:4
- The time needed to login and make a transaction: 15 second login, 35 second transaction
- Amount of data to remember: pincode (username e.g. accountnumber or accountholder)
- The number of physical items to keep (physical hardware token and/or token reader): 2 (token and mobile phone)
- The number of different systems used simultaniously: 3
- The familiarity with the solutions (by other sites / banks) [low, medium, high]: high
- Do customers need training / education in using it: no
- Calls into the Call Centre (user does not understand what to do / user has technical difficulties) [low, medium, high]: low
- Learning Curve [low,medium,high]: low
- Different behaviour compared to ”normal” Internet browsing: 0
- Necessary software installations on customer PC: none
- Operating System / browser dependency: no
- Device dependancy: yes (token and mobile phone)

- Is the solution ”perceived” to be secure: yes
- Ability to adapt the software/solution to the ”look and feel” of the bank: needs no changes

Security:

- Number of attacks it does not counter: User ignorance
- Degree of difficulty to perform possible attacks: very hard
- User skill-level/awareness dependence [0-3]: 1
- Maturity [0-4]: 4

Conclusion:

- User convenience:8,4,1,2,3,0,0,0,0,0,0,0,1,0,0 [19]
- Security:1,0,1,0 [2]

Connected ABN AMRO Model:
User convenience:

- The number of steps / operations (for the customer): login:4, transaction:4
- The time needed to login and make a transaction: 15 second login, 35 second transaction
- Amount of data to remember: pincode (username e.g. accountnumber or accountholder)
- The number of physical items to keep (physical hardware token and/or token reader): 1 (token)
- The number of different systems used simultaniously: 2
- The familiarity with the solutions (by other sites / banks) [low, medium, high]: medium
- Do customers need training / education in using it: no
- Calls into the Call Centre (user does not understand what to do / user has technical difficulties) [low, medium, high]: low
- Learning Curve [low,medium,high]: low
- Different behaviour compared to ”normal” Internet browsing: 0
- Necessary software installations on customer PC: none

- Operating System / browser dependency: no
- Device dependancy: yes (token)
- Is the solution ”perceived” to be secure: yes
- Ability to adapt the software/solution to the ”look and feel” of the bank: needs no changes

Security:

- Number of attacks it does not counter: User ignorance
- Degree of difficulty to perform possible attacks: very hard
- User skill-level/awareness dependence [0-3]: 0
- Maturity [0-4]: 3

Conclusion:

- User convenience:8,4,1,1,2,1,0,0,0,0,0,0,1,0,0 [18]
- Security:1,0,0,1 [2]

Disconnected ABN AMRO model:
User convenience:

- The number of steps / operations (for the customer): usb-stick insertion:1, login:3, transaction:4
- The time needed to login and make a transaction: 15 second login, 25 second transaction
- Amount of data to remember: pincode (username e.g. accountnumber or accountholder)
- The number of physical items to keep (physical hardware token and/or token reader): 1 (token)
- The number of different systems used simultaniously: 2
- The familiarity with the solutions (by other sites / banks) [low, medium, high]: high
- Do customers need training / education in using it: no
- Calls into the Call Centre (user does not understand what to do / user has technical difficulties) [low, medium, high]: low

- Learning Curve [low,medium,high]: low
- Different behaviour compared to "normal" Internet browsing: 0
- Necessary software installations on customer PC: software
- Operating System / browser dependency: yes
- Device dependancy: yes (token)
- Is the solution "perceived" to be secure: yes
- Ability to adapt the software/solution to the "look and feel" of the bank: needs no changes

Security:

- Number of attacks it does not counter: MitB, Phishing, Pharming, OS malware
- Degree of difficulty to perform possible attacks: easy
- User skill-level/awareness dependence [0-3]: 0
- Maturity [0-4]: 3

Conclusion:

- User convenience:8,3,1,1,2,0,0,0,0,0,1,1,1,0,0 [18]
- Security:5,3,0,2 [10]

Model 1: Thin server-side virtual machines
User convenience:

- The number of steps / operations (for the customer): login:4, transaction:3
- The time needed to login and make a transaction: 10 application download, 15 second login, 25 second transaction(strongly depends on internet connection speed)
- Amount of data to remember: pincode (username e.g. accountnumber or accountholder)
- The number of physical items to keep (physical hardware token and/or token reader): 1 (token)
- The number of different systems used simultaniously: 2
- The familiarity with the solutions (by other sites / banks) [low, medium, high]: medium

- Do customers need training / education in using it: yes
- Calls into the Call Centre (user does not understand what to do / user has technical difficulties) [low, medium, high]: low
- Learning Curve [low,medium,high]: low
- Different behaviour compared to "normal" Internet browsing: 0
- Necessary software installations on customer PC: Java JRE or another programming language
- Operating System / browser dependency: no
- Device dependancy: yes (token)
- Is the solution "perceived" to be secure: yes
- Ability to adapt the software/solution to the "look and feel" of the bank: application appearance needs to represent bank, but needs no changes to website appearance.

Security:

- Number of attacks it does not counter: OS malware (Phishing, Pharming is very difficult because of the performance load)
- Degree of difficulty to perform possible attacks: very hard
- User skill-level/awareness dependence [0-3]: 0
- Maturity [0-4]: none existent for banking purposes [3]

Conclusion:

- User convenience:7,4,1,1,2,1,1,0,0,0,1,0,1,0,1 [20]
- Security:1,0,0,1 [2]

Model 2: Hardened browser against Man-in-the-Browser attacks
User convenience:

- The number of steps / operations (for the customer): browser modification:1, login:4, transaction:3
- The time needed to login and make a transaction: 15 second login, 25 second transaction
- Amount of data to remember: pincode (username e.g. accountnumber or accountholder)

- The number of physical items to keep (physical hardware token and/or token reader): 1 (token)
- The number of different systems used simultaniously: 2
- The familiarity with the solutions (by other sites / banks) [low, medium, high]: high
- Do customers need training / education in using it: yes (but very little)
- Calls into the Call Centre (user does not understand what to do / user has technical difficulties) [low, medium, high]: low
- Learning Curve [low,medium,high]: low
- Different behaviour compared to ”normal” Internet browsing: 1
- Necessary software installations on customer PC: none
- Operating System / browser dependency: yes (Internet Explorer, IE disables BHO’s, or browsers that do not support Add-ons dependent)
- Device dependancy: yes (token)
- Is the solution ”perceived” to be secure: yes
- Ability to adapt the software/solution to the ”look and feel” of the bank: needs no changes

Security:

- Number of attacks it does not counter: Phishing E-Mail/IM, Pharming, OS malware
- Degree of difficulty to perform possible attacks: hard (people do not give in to phishing emails a lot anymore)
- User skill-level/awareness dependence [0-3]: 3
- Maturity [0-4]: 3 (not a 4 because of the immaturity of InPrivate mode)

Conclusion:

- User convenience:8,3,1,1,2,0,1,0,0,1,0,1,1,0,0 [19]
- Security:3,1,3,1 [8]

Model 3: Trusteer against Man-in-the-Browser attacks
User convenience:

- The number of steps / operations (for the customer): login:4, transaction:3
- The time needed to login and make a transaction: 15 second login, 25 second transaction
- Amount of data to remember: pincode (username e.g. accountnumber or accountholder)
- The number of physical items to keep (physical hardware token and/or token reader): 1 (token)
- The number of different systems used simultaniously: 2
- The familiarity with the solutions (by other sites / banks) [low, medium, high]: medium
- Do customers need training / education in using it: yes (use of Trusteer)
- Calls into the Call Centre (user does not understand what to do / user has technical difficulties) [low, medium, high]: medium
- Learning Curve [low,medium,high]: medium
- Different behaviour compared to "normal" Internet browsing: 1
- Necessary software installations on customer PC: Trusteer
- Operating System / browser dependency: yes (Trusteer Rapport currently supports Windows XP, Windows Vista, and Windows 7 running Internet Explorer 6, 7, 8 or Firefox 2, 3)
- Device dependancy: yes (token)
- Is the solution "perceived" to be secure: yes
- Ability to adapt the software/solution to the "look and feel" of the bank: Trusteer's appearance can not be changed, but user experience stays the same.

Security:

- Number of attacks it does not counter: OS malware (Trusteer does counter some OS malware but not all)
- Degree of difficulty to perform possible attacks: very hard
- User skill-level/awareness dependence [0-3]: 3
- Maturity [0-4]: 1 (not a 2 because of the immaturity and dependents of Trusteer)

Conclusion:

- User convenience:7,3,1,1,2,1,1,1,1,1,1,1,1,0,0 [22]
- Security:1,0,3,3 [7]

Model 4: USB-application
User convenience:

- The number of steps / operations (for the customer): usb-stick insertion:1, login:1, transaction:3
- The time needed to login and make a transaction: 5 second usb-stick insertion, 5 second login, 25 second transaction
- Amount of data to remember: username/password
- The number of physical items to keep (physical hardware token and/or token reader): 2 (usb-stick and token)
- The number of different systems used simultaniously: 3 (token/device, usb-stick and computer)
- The familiarity with the solutions (by other sites / banks) [low, medium, high]: medium
- Do customers need training / education in using it: yes (use of usb-stick application)
- Calls into the Call Centre (user does not understand what to do / user has technical difficulties) [low, medium, high]: medium (make a step-by-step application guide)
- Learning Curve [low,medium,high]: medium
- Different behaviour compared to ”normal” Internet browsing: 0
- Necessary software installations on customer PC: Java JRE or another programming language
- Operating System / browser dependency: no
- Device dependancy: yes (usb access)
- Is the solution ”perceived” to be secure: yes
- Ability to adapt the software/solution to the ”look and feel” of the bank: Develop usb-stick application in bank ”look and feel”.

Security:

- Number of attacks it does not counter: OS malware
- Degree of difficulty to perform possible attacks: hard
- User skill-level/awareness dependence [0-3]: 0
- Maturity [0-4]: 3

Conclusion:

- User convenience:5,2,1,2,3,1,1,1,1,0,1,0,1,0,1 [20]
- Security:1,1,0,1 [3]

Model 5: Processor-based USB-application
User convenience:

- The number of steps / operations (for the customer): login:1, transaction:2
- The time needed to login and make a transaction: 5 second login, 25 second transaction
- Amount of data to remember: username/password
- The number of physical items to keep (physical hardware token and/or token reader): 1 (IBM usb-stick)
- The number of different systems used simultaniously: 2
- The familiarity with the solutions (by other sites / banks) [low, medium, high]: low
- Do customers need training / education in using it: yes (learn to use usb-stick)
- Calls into the Call Centre (user does not understand what to do / user has technical difficulties) [low, medium, high]: medium
- Learning Curve [low,medium,high]: medium
- Different behaviour compared to ”normal” Internet browsing: 0
- Necessary software installations on customer PC: 0
- Operating System / browser dependency: no
- Device dependancy: yes (IBM usb-stick)
- Is the solution ”perceived” to be secure: yes
- Ability to adapt the software/solution to the ”look and feel” of the bank: IBM usb-stick has its own ”look and feel”.

Security:

- Number of attacks it does not counter: OS malware
- Degree of difficulty to perform possible attacks: very hard
- User skill-level/awareness dependence [0-3]: 0
- Maturity [0-4]: 3

Conclusion:

- User convenience:3,1,1,1,2,2,1,1,1,0,0,0,1,0,1 [15]
- Security:1,0,0,1 [2]