An Exprimental Investigation of the Usability of Transaction Authorization in Online Bank Security Systems

Mohammed AlZomai **Bander AlFayyadh** **Audun Jøsang** **Adrian McCullagh**

Queensland University of Technology
Brisbane, Australia
{alzomaim, b.alfayyadh}@isi.qut.edu.au and {a.josang, a.mccullagh}@qut.edu.au

Abstract

Security for online banking has changed considerably during the relatively short period that online banking has been in use. In particular, authentication and identity management in the early implementations were, and sometimes still are, vulnerable to various attacks such as phishing. Current state-of-the art solutions include methods for re-authenticating users via out-of-band channels for each transaction. This paper describes a security investigation of this type of solution. The investigation concludes that it protects against certain attacks while still being vulnerable to other obvious attacks. In the near future, it is expected that the remaining vulnerabilities will be exploited as the attackers get more sophisticated. Possible ways of protecting against these future attacks are outlined.

Keywords

Security, usability, identity management, authentication, authorization, online banking.

1 Introduction

Identity management is normally interpreted as the management of users' credentials and how they access a system. Identity management systems can thus be seen as consisting of an authentication part which is used to verify the correctness of an entity's claim to identity, and an access control part which grants access to applications and resources residing on a system or in a network. The authentication and access control parts are often tightly integrated.

With online services there are two types of authentications; user authentication and data origin authentication. User authentication is the process of verifying the digital identity of an entity. It is a way of ensuring that users are who they claim to be when they access systems. On the other hand, data origin authentication is to prove that the source of data is as claimed. It is the verification that data has not been tampered with in transit (data integrity) and that it originated from the expected sender (authenticity).

In online banking, data origin authentication is important. Although the user has logged on from a specific client terminal and has been authenticated at the start of a session, this in itself does not guarantee that every data packet originating from the client terminal is the intentional result of user actions. For example, a Trojan[1] application could initiate online bank transactions from the client terminal without the user's consent or knowledge. Data origin authentication can theoretically eliminate this threat by authenticating the transaction request itself.

As a response to the growing threats to online banking security (such as phishing and fraud) and to enhance the security, online bank systems usually implement special methods for authentication. These methods allow the authentication process at the transaction level by involving the user more in the security system having him/her confirming every transaction. User authentication alone is insufficient given the vulnerability of the standard client terminal and the relatively high risk of online bank transactions.

A typical method for data origin authentication is to use an OTP (One-Time-Password) for each transaction. Banks can implement this by issuing special hardware tokens that can generate one-time authorisation codes. An OTP token is a password generator device with an LCD screen which displays a pseudo-random number consisting of 6 or more alphanumeric characters (Studies showed that capacity of short term human memory load is normally 7 +/- 2 items (Miller 1956)). The pseudo-random number changes when clicking a button on the token or at a specific time interval such as every 30 or 60 seconds. The device is synchronized with a peer OTP generator on the service provider side and both tokens generate the same sequence of numbers. The OTP must be copied manually from the token to the client terminal.

Another method for data origin authentication used by online bank systems is based on sending OTP with SMS messages to the user's mobile phone for each financial transaction. As for the OTP token, the user must manually copy the OTP from the mobile phone screen to the client terminal in order to confirm each financial transaction.

The term *"authorization code"* is often used by online banks to denote the OTP, because it is required to "authorize" transactions. We will use this term instead of OTP when discussing online bank security below.

A good identity management system should address all identity management related aspects such as authentication, authorization, privacy and usability (de Clercq & Rouault 2004, Casassa Mont et al. 2002). This paper describes a practical experiment aimed at investigating the security and usability of SMS-based data origin authentication for authorizing online banking transactions.

[1] A Trojan is a malicious software application that is not controlled by the owner of the computer.

2 The SMS Authorization Scheme

2.1 Architecture

The main advantage of the SMS authorization scheme is that SMS messages sent from the bank to the user's mobile phone pass through the cellular network, which is separate and independent from the Internet. By verifying the authorization code received from the client terminal, the bank can conclude that the user received the SMS message through the cellular network, read it and submitted it through the Internet. This is then interpreted as a genuine intent to submit the transaction. The security of this scheme is based on the assumption that it is difficult for an attacker to steal the user's personal mobile phone and to attack the cellular network (Jøsang et al. 2007a). The scenario is illustrated in Fig.1 and Table 1.

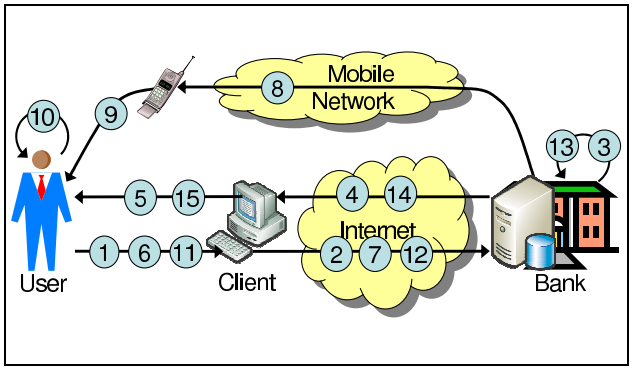

Figure 1: Authorising bank transactions via SMS

Msg ♯	Message description
1	Produce Login Id and authentication token
2	Transmit Login Id and authentication token
3	Verify Login Id and authentication token
4	Transmit service options
5	Present service options
6	Transaction request
7	Transmit transaction request
8	SMS message with authorisation code
9	Read SMS message
10	Verify amount and bank account number
11	Copy authorisation code
12	Transmit authorisation code
13	Verify authorisation code
14	Transmit transaction confirmation
15	Present transaction confirmation

Table 1: Messages in SMS bank security protocol

The SMS authorisation code is computed as a function of the origin and destination accounts, as well as the amount. It typically consists of a number of digits that can be copied manually from the mobile phone to the client terminal without too much effort. A typical SMS authorization message is illustrated in Fig.2.

2.2 Security Analysis

When the authorization code is typed on the client terminal and sent to the online bank, the transaction will be executed. Assuming that an attacker changes the amount and/or the destination account number, e.g. by a Trojan program on the client terminal, the modified amount and

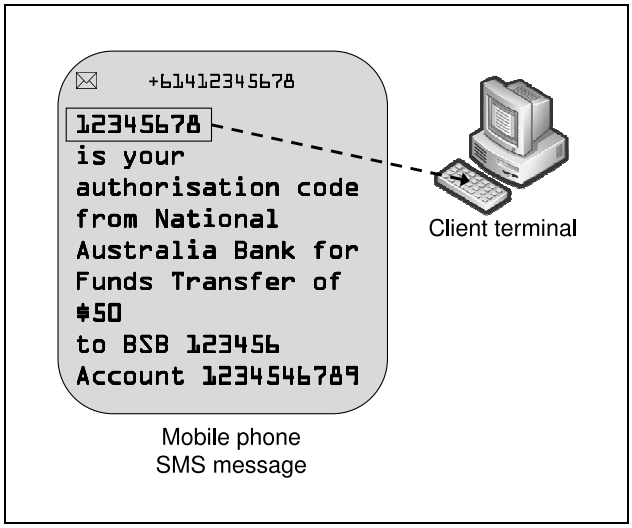

Figure 2: Example SMS message with authorisation code

account number will appear in the SMS message. It is assumed that the correctness of the amount and of the destination account number is verified by the user when copying the authorisation code from the SMS message. If a user victim fails to notice that the bank account number in the SMS message is not the same as the intended account number, and submits the authorisation code through the client terminal, the attack will succeed.

Assuming that the user verifies the correctness of the amount and of the bank account number in the SMS message, this scheme is secure against attacks on the client terminal, and is in fact independent of the security of the client terminal. This represents a considerable security improvement. It is possible to leverage the use of SMS messages in this way because the amount of crucial data is sufficiently small to be communicated in a single SMS message.

This scheme assumes that the mobile terminal can be trusted, i.e. that no attacker is able to take over the control of a mobile terminal. This assumption can not be made for the standard client terminal. If it were possible to take over the control of the mobile terminal, an attacker could change the SMS message, and present the expected amount and the bank account number, so that the SMS message that the user reads is not the same as the SMS message that the bank sent.

The scheme depends to some degree on the security of the mobile phone networks, and it assumes that no attacker is able to modify SMS messages sent from the bank to the user. Even if interception and cryptanalysis of SMS messages sent over the air were possible, it requires that the attacker is physically present in the same base station coverage area, and this excludes attacks from anywhere in the world.

However, SMS authentication schemes may be vulnerable to delay and unreliable mobile SMS delivery. Availability is a fundamental security goal which means that data and resources must be accessible when needed by an authorized user. The SMS authentication scheme may violate the availability principle since SMS messages traveling across different mobile networks may not arrive in a timely fashion, causing service denial (Jiang 1998). The problem is amplified manifold when SMS messages travel between different mobile network operators in different countries.

3 Security Usability Considerations

Security systems for online banking need to provide adequate security usability and should have a simple and intuitive user interface. The system should not only be designed to satisfy service provider requirements, but must also satisfy user requirements, otherwise it will lead to inconvenience and poor usability for users. With poor usability and a poor user interface with regard to security, the system will have poor security.

Usability of security is an extremely important, but still poorly understood, element of IT security. One of the earliest studies in this area is the experimental investigation of the usability of PGP by Whitten and Tygar (Whitten & Tygar 1998, 1999).

A set of general security usability principles was defined in (Jøsang et al. 2007a,b). These principles distinguish between two types of user involvement with security applications.

- A *security action* is when users are required to produce information and security tokens, or to trigger some security relevant mechanism. For example, typing and submitting a password is a security action.
- A *security conclusion* is when users observe and assess some security relevant evidence in order to derive the security state of systems. For example, observing a closed padlock on a browser, and concluding that the communication is protected by SSL is a security conclusion.

Usability principles related to security actions and security conclusions are described below.

1. Security Action Usability Principles
   (a) The users must understand which security actions are required of them.
   (b) The users must have sufficient knowledge and the practical ability to make the correct security action.
   (c) The mental and physical load of a security action must be tolerable.
   (d) The mental and physical load of making repeated security actions for any practical number of transactions must be tolerable.
2. Security Conclusion Usability Principles
   (a) The user must understand the security conclusion that is required for making an informed decision. This means that users must understand what is required of them to support a secure transaction.
   (b) The system must provide the user with sufficient information for deriving the security conclusion. This means that it must be logically possible to derive the security conclusion from the information provided.
   (c) The mental load of deriving the security conclusion must be tolerable.
   (d) The mental load of deriving security conclusions for any practical number of service access instances must be tolerable.

While the mental load of verifying the correct amount and destination account specified by the SMS message is probably acceptable for a single transaction, the repeated process of verifying the same can be quite tedious leading to user apathy, thereby violating security usability principle 2d. It has been noted that when faced with a frustrating security task, users may usually bypass or ignore that task (Balfanz et al. 2004, Sasse 2003, Adams & Sasse 1999). In order to determine the usability of such schemes an investigation was needed. The outcome of such a study is potentially very important because it will determine whether the user can be made liable for errors made when using such systems.

In the next sections, we will describe a practical experiment which simulated an online bank and test how participants in the experiment where able to correctly use the SMS authorization scheme for authorizing funds transfer transactions.

4 Experiment Design

4.1 Objectives

The goal of the experiment was to examine the usability of the SMS authentication scheme. In other words, we investigated whether users are able to perform the extra tasks in a satisfactory manner, and that executing these tasks does not introduce new security vulnerabilities. It is important to know whether users are able to fulfill these tasks, because it tells us whether it is reasonable to make users liable for transactions based on this security technique. Banks would normally assumed that users are responsible for transactions authorized with the authorization code. However, if a significant proportion of users are unable to use the method correctly, this assumption would be unreasonable and should be reassessed by the banks.

The experiment studied customers interaction with an online bank that uses the SMS authorization code scheme described above. We asked participants to play the role of customers and perform a number of financial transactions using our simulated virtual online bank. Their reaction to security attacks was monitored and analyzed. The analysis would show whether it is reasonable to make customers liable for errors made when using the system.

4.2 Ethical Considerations

The experiment was conducted during June and July 2007 at QUT (Queensland University of Technology) and involved human participation. The experiment was reviewed and approved by the QUT Research Ethics Committee. There were no risks associated with participation in the experiment. A Web page with a consent form was presented to participants before starting the experiment. Participants were asked to click a button labeled "Participate" to confirm their agreement to participate. The participants' email addresses were required as part of the experiment. The participants were informed that it would be kept confidential and would be deleted after completion of the experiment.

4.3 Participant Recruitment

The participants were recruited by sending out invitation emails. We obtained permissions to use several email distribution lists for sending out the invitations. The subject field in the invitation email said: *"Invitation: Online*

banking security experiment". The email body had the following content:

> *The Information Security Institute at QUT is running an experiment on the usability of online banking security, and you are invited to participate. You will find it fun and interesting to play with our simulated online bank. It will only take a few minutes. We would like you to transfer virtual money to different bank accounts imagining that you are using your real bank account and money. We hope you can make at least 10 transactions. It is important that you take the same security precautions as you would with your real online bank account. This means that you should cancel any transaction where you notice something suspicious, because it could indicate a security attack. The list of bank accounts is provided below.*
>
> *For each transaction, you will do the following:*
>
> 1. *Start a new transaction by filling in the "New Transaction" web page and clicking "Submit". You will receive an authorization code by email.*
> 2. *Fetch the email and verify that the transaction details are correct.*
> 3. *Confirm the transaction in case the details are correct. To confirm, copy the authorization code from the email to the Web page, and click "Confirm". Alternatively cancel the transaction if you think there is something wrong. To cancel, simply click "Cancel".*
>
> *Please visit the URL below to start the experiment.*
>
> *http://www.isi.qut.edu.au/people/alzomaim/bank/consent.htm*
>
> *List of destination accounts to be used*
>
> | *Destination accounts: (only these possible)* | *Suggested amounts: (other amounts possible)* |
> |---|---|
> | *30263142* | *$5000* |
> | *30263155* | *$500* |
> | *30263157* | *$5500* |
> | *30263143* | *$55000* |
> | *30263158* | *$50000* |
> | *30263145* | *$4400* |
> | *30263149* | *$44000* |
> | *30263150* | *$440* |
> | *30263144* | *$44400* |
> | *30263156* | *$400* |
>
> *We ask you to execute at least 10 transactions. Multiple transfers to the same account are possible.*
>
> *Thank you,*
> *Note: Approval from the QUT ethics committee has been obtained. Your email address will be kept confidential and will be deleted after completion of the experiment.*

Other participants were recruited by personal contact providing them with a hard copy of a document identical to the above email.

5 System Design

We developed the simulated online bank and asked participants to execute a number of financial transactions. Some of the transactions were corrupted to simulate attacks. The transaction records showed whether participants were able to notice the attacks and cancel these transactions.

We expected that the average participant would not be willing to provide us with their mobile phone numbers. Email messages provides the same functionality as SMS messages for the purpose of transaction authorisation. Also, we were not interested in investigating the technical security aspects of SMS based authorization. For these reasons we decided to base the authorization of transactions on email messages instead of SMS messages.

We developed a real Web system that simulated a virtual online bank consisting of two parts, the server front end that provides the http interface, and the server back end that handles the database (see Fig.3).

Both the front end and the back end were implemented using PHP (a programming language designed to build dynamic websites). The Bank database at the back end was a relational database designed, implemented and accessed using the MySQL relational database system.

5.1 Server Front End

The server front end was hosted on the Internet server of the Information Security Institute (ISI) at Queensland University of Technology (QUT). This server presented the simulated online bank interface where users could execute financial transactions. The user interface consisted of two web pages entitled *"New transaction web page"* and *"Confirmation web page"* that are described in more details below.

5.1.1 New Transaction Web Page

The "New Transaction" web page (see Fig.4) allowed users to initiate new transactions. The web page was designed to do the following functions:

- Display the web page content in the browser window allowing users to enter transaction information.
- Validate entered data format which included account number, amount and email address.
- Send entered transaction information to the bank server.

This page contained the following fields:

- Destination Account: An eight digit destination account number.
- Amount: Amount to be transferred
- E-Mail address: Where the authorization code is to be sent.

A *'Submit'* button triggered the transfer of the entered information to the virtual bank server and took the user to the Confirmation Web Page.

5.1.2 Confirmation Web Page

The 'Confirmation' web page (see Fig.5) allowed users to confirm and complete an initiated transaction. It was designed to do the following:

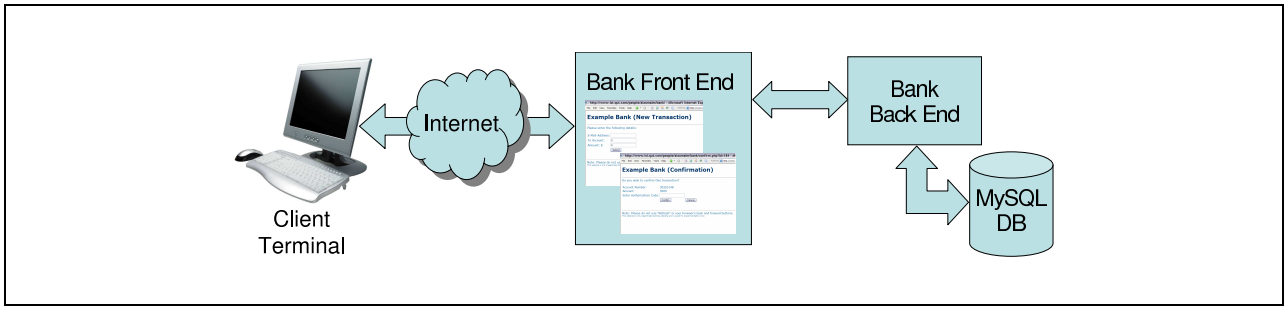

Figure 3: Simulated Online bank Model

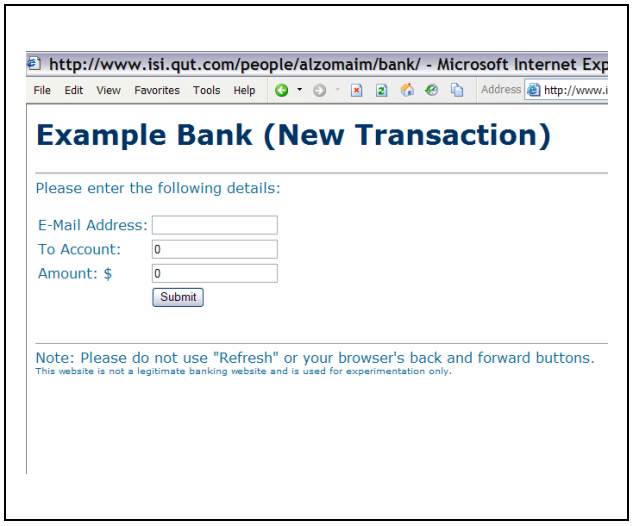

Figure 4: New Transaction Web Page

- Display the web page content in the browser window allowing users to enter the authorization code.
- Send the entered authorization code to the bank server.
- Display the transaction confirmation information.

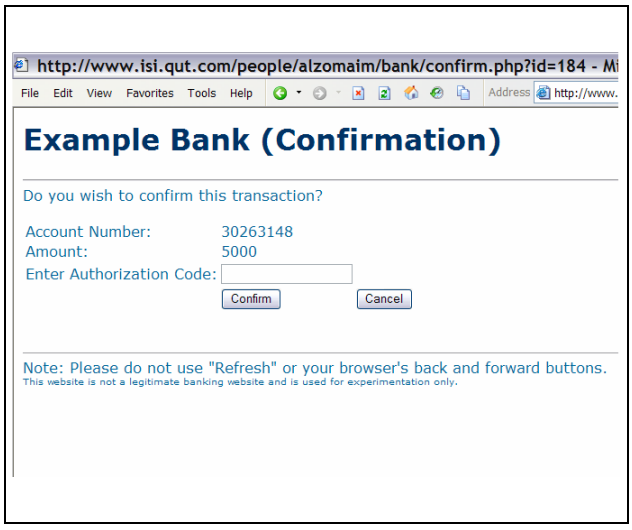

Figure 5: Confirmation Web Page

The page displayed the transaction information and contained a single field for the authorization code to be entered. The page contained a *'Confirm'* button used to confirm the transaction by sending the authorization code to the virtual bank server and a button labeled *'Cancel'* to cancel the transaction. If the user chose to confirm the transaction, a link to the 'New Transaction web page' was displayed allowing him or her to do another transaction. If the user chose to cancel the transaction, two links were displayed; one to do another transaction and another to return the user back to the confirmation page.

5.2 Server Back End

Most of the functionality of the simulated online bank model was executed in the server back end which was designed to do the following main functions:

- Generate authorization codes in the format of six-digit hex numbers. Each code resulted from applying the hash function SHA1[2] to the transaction information and a random number.
- Send transaction information to the 'Confirmation' web page.
- Email the authorization code and transaction information to the user email address.
- Verify received authorization code for a particular transaction.
- Generate security attacks by alteration of the transaction details in selected transactions.

The Bank database connected to the back end was a relational database which was designed, implemented and accessed using the MySQL relational database system. All user interactions with simulated online bank when executing financial transactions were stored in the bank database.

5.3 Participant Tasks

Participants were provided with instructions to complete a number of bank transactions using our simulated online bank. To complete each transaction, each participant had to do the following steps:

- Start a new transaction by filling the fields in the 'New Transaction' web page and then clicking the *'Submit'* button.
- Check email sent by the virtual bank server to validate transaction information and copy authorization code.
- Complete the transaction by entering the authorization code in the 'Confirmation' web page and clicking the *'Confirm'* button or alternatively cancel the transaction by clicking the *'Cancel'* button.

[2]Secure Hash Algorithm. A function that produces a fixed-length digital representation (known as a digest) of an input data sequence of any length.

We provided participants with a list of acceptable destination accounts and suggested amounts to be used. We also asked participants to complete at least 10 transactions. All transactions and data collected from user interactions with this experiment were recorded and stored in the database on the server back end side. The scenario of the experiment is illustrated in Fig.6 and Table 2.

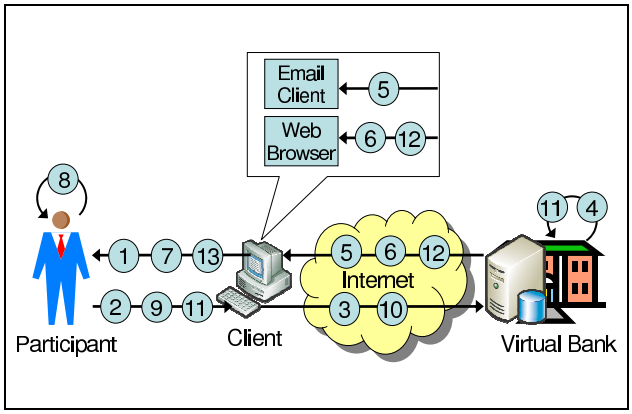

Figure 6: Authorising bank transactions via Email

Step ♯	Description
1	Present 'New Transaction' Web page.
2	Enter transaction information.
3	Transmit transaction information.
4	Generate authorisation code.
5	Email trans. info and authorisation code.
6	Transmit Confirmation Web page.
7	Present Confirmation Web page.
8	Verify received transaction information.
9	Enter authorisation code/cancel transaction.
10	Transmit authorisation code.
11	Verify authorisation code.
12	Transmit transaction confirmation
13	Present transaction confirmation

Table 2: Virtual online bank scenario

5.4 Simulated Security Attacks

Transaction authorization with SMS messages has been introduced by online banks in response to various security attacks such as phishing and man-in-the-middle attacks. While technically elegant, it puts extra cognitive burden on users because they have to handle two devices, and relate information between them. In the experiment, simulated attacks were executed during online bank transactions, and we observed whether users were able to notice that they were being attacked.

To conduct the test, we asked participants to execute a number of financial transactions. Some of those transactions were attacked by alteration of the destination account. The destination account was altered after the participant had submitted the initial transaction request. This resulted in an email containing the altered transaction information together with authorization code being sent to the participant's email address.

We implemented two types of security attacks; the *"obvious attack"* which was easy to discover, and the *"stealthy attack"* which was difficult to discover.

- **Obvious Attack Type**
  The obvious attack was designed so that it would be easy for participants to notice the alteration in the transaction details. This was done by altering five out of eight digits of the destination account number.
- **Stealthy Attack Type**
  The stealthy attack was designed such that it would be difficult for participants to notice the alteration in the transaction details. This was done by altering only one digit in the destination account number.

By observing whether participants canceled the attacked transactions we could determine whether participants noticed the obvious and the stealthy alterations to the destination account number.

5.5 Phase Shift of Attacks

To execute our security attacks equally over transactions, we assigned an attack phase to each participant in the following manner: Participant nr. 1 was assigned attack phase 1, participant nr. 2 was assigned attack phase 2, etc. until the tenth participant was assigned attack phase 10. Then participant nr. 11 was assigned attack phase 1 and so on.

Attack phases determined when the stealthy attack was going to occur. For example, if attack phase for a particular participant was 7, then the stealthy attack would occur on the seventh transaction for that particular participant. The obvious attack occurs on a transaction number calculated by ((attack phase + 5) mod 10). For example, if a participant was assigned an attack phase 7, then the obvious attack would occur on his or her second transaction (i.e. (7 + 5) mod 10 = 2).

Participants with attack phases 1-5 would face stealthy attack first while those with attack phases 6-10 would face obvious attack first. Table 3 shows when both types of attacks would occur for each attack phase.

Phase shift	Transaction # attacked	
	Stealthy	Obvious
P.s. 1	#1	#6
P.s. 2	#2	#7
P.s. 3	#3	#8
P.s. 4	#4	#9
P.s. 5	#5	#10
P.s. 6	#6	#1
P.s. 7	#7	#2
P.s. 8	#8	#3
P.s. 9	#9	#4
P.s.10	#10	#5

Table 3: Phase shift of attacks

6 Participant Demographics

The majority of participants in the experiment were staff and students at Queensland University of Technology. The participant recruitment process was to send a participation invitation email to a certain distribution list and start receiving responses from participants and collecting data. After a period when no new participants had been observed, another email to another distribution list was sent out and so on.

From the recruitment procedure we followed, we can classify participants into the following groups:

- **QUT Participants**

- 10 participants (11%) were PhD research students at Information Security Institute (ISI).
- 5 participants (5%) were staff at the Faculty of Information Technology.
- 58 participants (64%) were undergraduate and master students at the Faculty of Information Technology.
- 5 participants (5%) were students at the Faculty of Education.
- 4 participants (4%) were researchers at Faculty of Law.

- **Non-QUT Participants** 10 non-QUT participants (11%) were contacted personally and non of them has a degree in Information Technology.

  The actual number of people responded to our invitation email and started working on the experiment was 116, but 24 participants were excluded for different reasons:

  - 3 participants with invalid email addresses.
  - 12 participants executed few transactions without being attacked.
  - 9 participants initiated transactions but never completed the confirmation process. We noticed that some email servers, like *HOTMAIL*, directed our invitation email to a spam folder.

Fig.7 summarizes the participant demographics.

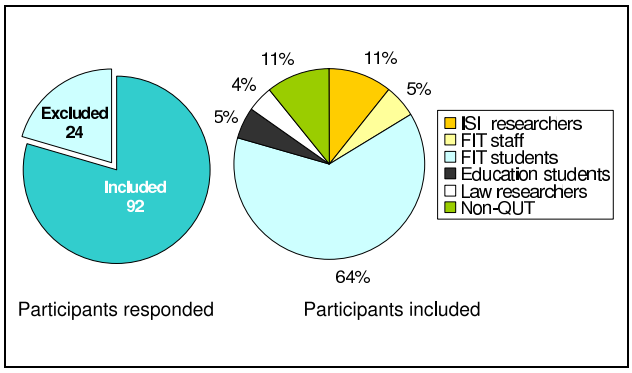

Figure 7: Participant Demographics

7 Results

As a result of the experiment, 92 participants executed a total of 734 transactions. When a participant interacted with the system, each transaction could be in one of two states:

1. An initiated transaction which was a funds transfer transaction a participant had started but not confirmed.
2. A complete transaction which was a financial transaction a participant had started and confirmed.

We classified experiment transactions into the following:

- Normal transaction: This was a completed normal unaltered transaction.
- Incomplete or error transaction: This was a normal unaltered transaction that participant initiated but did not complete.
- Successfully attacked transaction: This was a completed transaction where the participant failed to notice the altered destination account number.
- Unsuccessfully attacked transactions (avoided attack): This was an altered transaction that was canceled by the participant.

There were 557 normal transactions, 49 incomplete transactions and 128 attacked transactions where 57 transactions were successfully attacked and 71 transactions were unsuccessfully attacked. See Table 4, Table 5 and Fig.8.

Transaction type	Count
Normal	557
Incomplete	49
Successful attacks	57
Avoided attacks	71
Total	734

Table 4: Overview of recorded transactions

Attack type	Avoided	Successful	Total
Obvious	42	11	53
Stealthy	29	46	75
Total	71	57	128

Table 5: Overall attack response

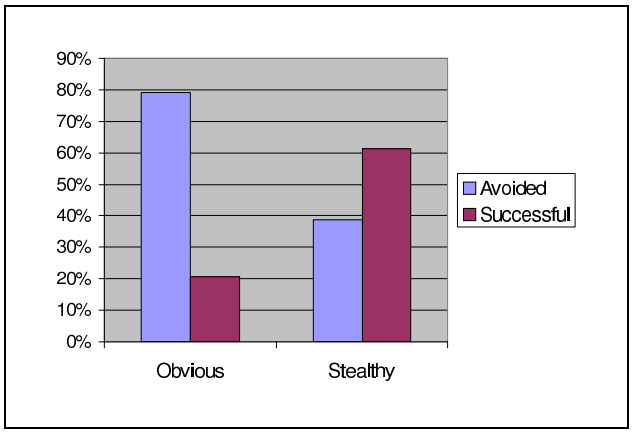

Figure 8: Overall attack response

7.1 Observations of Obvious Attacks

The total number of transactions attacked by the obvious attack was 53. Participants were able to discover and cancel 42 attacked transactions of this type. This means that participants were able to avoid 79% of the obvious attacks. Only 11 out of 53 attacked transactions of this type were successful, which translated into 21% successful obvious attacks. See Table 5 and Fig.8.

7.2 Observations of Stealthy Attacks

The total number of transactions attacked by the stealthy attack was 75. Participants were able to discover and cancel only 29 attacked transactions of this type. This means that participants were able to avoid 39% of the stealthy attacks. 46 out 75 attacked transactions of this type were successful which translates into 61% successful stealthy attacks. See Table 5 and Fig.8.

7.3 Attack Phases and Attack Type

Some of our participants had been attacked first by obvious attack while others faced the stealthy attack first. We found out the number of successful and avoided attacks varied depending on which type of attack occurred first.

With obvious attack occurring first, the number of successful obvious attack was 9 out of 32 while there were 17 out of 29 successful stealthy attacks. See Table 6 and Fig.9.

Attack type	Avoided	Successful	Total
Obvious	23	9	32
Stealthy	12	17	29
Total	35	26	61

Table 6: Obvious attack first

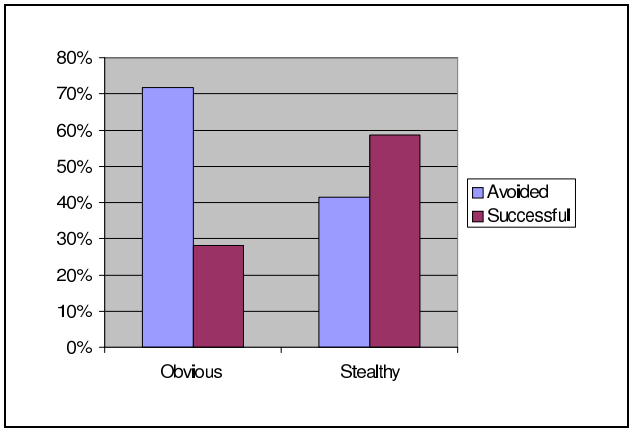

Figure 9: Obvious attack first

On the other hand, when stealthy attack occurred first the number of successful stealthy attack was 29 out of 46 and the number of successful obvious attacks was 2 out of 21. See Table 7 and Fig.10.

Attack type	Avoided	Successful	Total
Obvious	19	2	21
Stealthy	17	29	46
Total	36	31	67

Table 7: Stealthy attack first

8 Discussion

The observations show that a significant proportion of users are unable to detect the attacks. As predicted, the obvious attacks were detected more frequently than the stealthy attacks. The detection rate of stealthy attacks depends to a certain degree on whether it was preceded by an obvious attack or not.

An interesting observation is that attacks are avoided significantly more often when occurring late in the users experience with the online bank.

For example obvious attacks were avoided in 72% of cases when occurring before stealthy attacks, and in 90% of the cases when occurring after stealthy attacks.

Similarly, stealthy attacks were avoided in 37% of the cases when occurring before obvious attacks, and in 41% of the cases when occurring after obvious attacks.

A possible explanation for this trend can be that users need to get a degree of experience with the system before

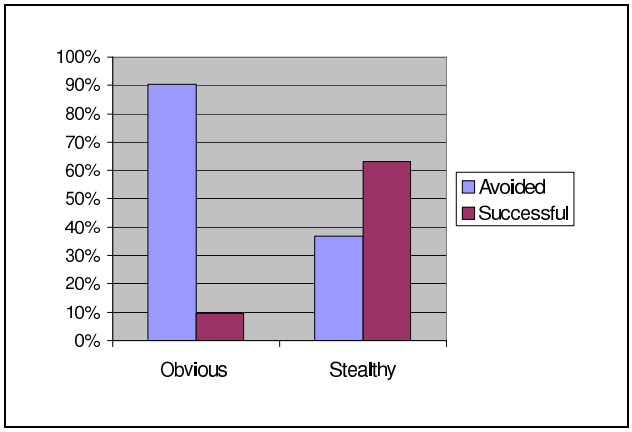

Figure 10: Stealthy attack first

they become sufficiently alert to detect and avoid attacks. In other words, it is possible that the novelty of the systems has the effect of diverting peoples attention.

It can be assumed that realistic attacks will require alteration of about half the digits in the destination account number, which translates into our obvious attacks. We will therefore base the discussion of security on the observed statistics of obvious attacks. Disregarding the order in which obvious and stealthy attacks occurred, it is worrisome to observe that obvious attacks were successful in 21% of the cases in general. The conclusion to be drawn from this, which is also the main conclusion of the experiment, is that this system is very insecure against attacks where the amount and destination account are changed in the authorization message. It is easy to imagine attack scenarios where this could happen.

8.1 Trojan Attack Scenario

Attackers can send out spam email which invites users to access a website that will install a Trojan on users client computer. This Trojan will observe activities on the client computer and get into action when the user starts an online bank session. When the user specifies a funds transfer transaction, the Trojan will alter the amount and destination account without displaying the alteration on the screen. The online bank will thus receive a transaction request with the false amount and destination account. Even when the transaction requires authorization via an SMS message, a significant percentage of users will fail to notice that the transaction details have been altered.

8.2 Man-In-the-Middle Attack Scenario

It is possible for an attacker to intercept the communication between the customer and the bank server and impersonate them both. The attacker could trick the customer into logging into the attacker's website, and masquerade as the real bank. This can for example happen through the attack commonly known as *pharming* (Berghel 2006, Madsen et al. 2005). This consists of poisoning the DNS[3]cache on the client terminal or local broadband router, so that the domain name of the genuine online bank corresponds to the IP address of the attacker's server in the poisoned DNS cache. With a poisoned DNS, the browser will connect to the attacker's server even though the customer manually types the correct domain name of the bank. DNS poisoning can easily be executed by luring the customer to access a malicious

[3]Domain Name Server

website loaded with JavaScripts and JavaApplets (Stamm et al. 2006, Tsow et al. 2006).

By obtaining customer's authentication details, the attacker can login to the legitimate bank website and act as a man-in-the-middle between the customer and the online bank. When the customer sends a transfer transaction request to the attacker website, the attacker can relay the similar altered transaction request (i.e. by changing the destination account number) to the real online bank. Upon receiving the altered transaction request, the online bank will then send an SMS message containing the authorization code and the false transaction details to the customer. Our observation shows that a considerable percentage of customers will not notice the alteration and will complete the confirmation process by sending the authorization code to the attacker; in this case, the attacker can relays the received authorisation code to the bank which will execute the altered transaction. The attack has succeeded!

9 Conclusion

The transaction authorization method based on SMS messages was introduced by banks in response to the now traditional phishing attacks, and this method is indeed effective in stopping such attacks. Unfortunately, it is expected that it is only a matter of time before the attacks get more sophisticated, such as for example through smart Trojans installed on the users client computers or through Man-In-the-Middle attacks.

It is worth considering that participants in our experiment may be less motivated to behave securely than they would in real life when dealing with actual bank accounts. From the point of view of the participants, there is no risk attached with performing the experiment whereas there are real risks involved in conducting transactions through their online bank. Actually, it would be impossible to create an environment with real risk so this was an unavoidable limitation in our experiment. However, our study has given a strong indication that the SMS transaction authorization method will be relatively vulnerable to the attacks we have described. According to our observations only about 79% of users would be able to avoid realistic attacks, which in our opinion represents an inadequate level of security for online banking. We predict that online banks will need to develop improved methods for ensuring the security and integrity of online banking when such attacks start occurring.

The security problem caused by the failure to notice that transaction details have been altered has more to do with usability than with technical security. A possible solution should therefore be based on an improvement in usability, and not necessarily on improving security mechanisms. However, it is beyond the scope of this study to propose a possible solution to the discovered security problem.

There will always be a trade-off between different goals when designing identity management solutions, and it is natural that the service and infrastructure providers will promote solutions that are cost effective relative to the assumed risk. However, as the threat picture changes, the solutions need to be adapted. We hope that this study will allow online banks and other online service providers to be better prepared for emerging risks.

References

Adams, A. & Sasse, M. A. (1999), 'Users are not the enemy', *Communications of the ACM* **42**(12), 40–46.

Balfanz, D., Durfee, G., Smetters, D. K. & Grinter, R. E. (2004), 'In search of usable security: five lessons from the field', *Security and Privacy Magazine, IEEE* **2**(5), 19–24.

Berghel, H. (2006), 'Phishing mongers and posers', *Commun. ACM* **49**(4), 21–25.

Casassa Mont, M., Bramhall, P., Gittler, M., Pato, J. & Rees, O. (2002), 'Identity Management: a Key e-Business Enabler', Hewlett-Packard Laboratories.

de Clercq, J. & Rouault, J. (2004), An Introduction to Identity Management, Technical report, Hewlett-Packard Company.

Jiang, H. (1998), 'Reliability, costs and delay performance of sending short message service in wireless systems', *Universal Personal Communications* **vol.2**, 1073–1077.

Jøsang, A., Alfayyadh, B., Grandison, T., Alzomai, M. & McNamara, J. (2007b), Security Usability Principles for Vulnerability Analysis and Risk Assessment, *in* 'The Proceedings of the Annual Computer Security Applications Conference (ACSAC'07)'.

Jøsang, A., Alzomai, M. & Suriadi, S. (2007a), Usability and Privacy in Identity Management Architectures, *in* 'The Proceedings of the Australasian Information Security Workshop'.

Madsen, P., Koga, Y. & Takahashi, K. (2005), Federated Identity Managemnt for Protecting users from Id theft, *in* 'Proceedings of the 2005 workshop on Digital Identity management'.

Miller, G. (1956), 'The magical number seven, plus or minus two: Some limits on our capacity for processing information', *The Psychological Review* **63**, 81–97.

Sasse, M. (2003), Computer Security: Anatomy of a Usability Disaster, and a Plan for Recovery, *in* 'Proceedings of the Conference on Human Factors in Computing Systems (CHI2003), (Workshop on Human-Computer Interaction and Security Systems)'.

Stamm, S., Ramzan, Z. & Jakobsson, M. (2006), TR641: Drive-By Pharming, Technical report, Indian University, Department of Computer Science.

Tsow, A., Jakobsson, M., Yang, L. & Wetzel, S. (2006), 'Warkitting: The drive-by subversion of wireless home routers', *Digital Forensic Practice* **vol.1**(no.3), 179 – 192.

Whitten, A. & Tygar, J. (1998), Usability of Security: A Case Study, Computer Science Technical Report CMU-CS-98-155, Carnegie Mellon University.

Whitten, A. & Tygar, J. (1999), Why Johnny Can't Encrypt: A Usability Evaluation of PGP 5.0, *in* 'Proceedings of the 8th USENIX Security Symposium'.