WHITE PAPER: SYMANTEC SECURITY RESPONSE

“Phishing In The Middle Of The Stream” - Today’s Threats To Online Banking

Candid Wüest
Symantec Security Response, Dublin

From the proceedings of the AVAR 2005 conference. Used with permission of the author.

“Phishing In The Middle Of The Stream” Today’s Threats To Online Banking

Contents

Abstract...5

Introduction... 5

Remote attacks... 6

Phishing.. 6

DNS attacks...8

DNS cache poisoning.. 8

DNS Hijacking.. 9

Interception...9

Local attacks... 9

Hybrid attacks...12

Protection methods.. 13

SMS challenge code...13

Image verification.. 15

Dynamic Security Skins.. 16

Counter phishing methods..17

Domain hashed passwords 17

PKI based software solution 18

PKI based hardware token..19

Contents (continued)

Comparison 21

Conclusion 22

References 24

About the Author 26

Abstract

Malicious applications that steal financial account information have increased dramatically over the last year, potentially resulting in a direct loss of hard currency to affected victims. While the primary target continues to be online financial systems, the methods used to gather the sensitive information vary. Attacks spread from simply spamming e-mails with links to fake web sites, which is also known as phishing, to Trojans that monitor attempts to log on to online account web services and then begin recording the pressed key strokes, take screen shots, or even redirect the whole network traffic to a malicious site.

This paper will discuss the methods that are used by online banking malware, analyze their chances of successfully stealing sensitive data, and show some of the defense techniques that have evolved to obviate their attacks.

Introduction

The use of the Internet has grown in our daily lives, many services are now available online. This new business market offers many opportunities for service providers, including financial institutions. It should not surprise anyone that wherever money is involved, villains appear trying to steal it. The same applies to online financial systems. The surprising part is the recent increase in the number and the evolution of their techniques. In May 2003 only around twenty Trojans that target financial services where reported in the wild, most of them with basic functionality. Two and a half years later the number of such malware has increased to nearly two thousand different variants [1]. Often skeptical people say that these are only theoretical attacks, which would never succeed. The growing number of examples proves the opposite, as in the case of an online bank heist in Korea in June 2005 [2]. A student with only average computer skills used a basic Trojan to steal $50,000 from other people's bank accounts.

Discovered in August 2003, PWSteal.Bancos.B [3] targeted account information from only five banks. Its newer version PWSteal.Bancos.T discovered in April 2005 [4], contained an astonishing list of 2764 monitored URLs from 59 different top-level domains. The corresponding web sites belong to online financial services and online commerce sites, ranging from small branch offices to international groups. This shows that these malware are no longer a small local phenomenon.

In February 2005, a Trojan named Trojan.Goldun.B [5] was found stealing logon credentials for an online payment service called e-gold. Downloaded through peer-to-peer networks or received by spam emails the Trojan found its way to the user's machine. When a user executed the deceptively named file, SecurityEgold.exe, the Trojan installed itself as a browser helper object and started monitoring the visited web sites for predefined URLs. All intercepted account information was posted via a PHP script on a domain controlled by the attacker. Due to a poorly configured web server the log file storing all the gathered data was accessible by anyone. A quick look at this log file confirmed a growing list of account numbers and corresponding passwords. Within an hour, 13 valid-looking sets of credentials were added by the script. This site was online for another 24 hours before being shut down. Only 5 days later the next variant of this Trojan family appeared in the wild. This suggests that the attacker may have obtained even more account data than they could subsequently use to empty the victims' online account.

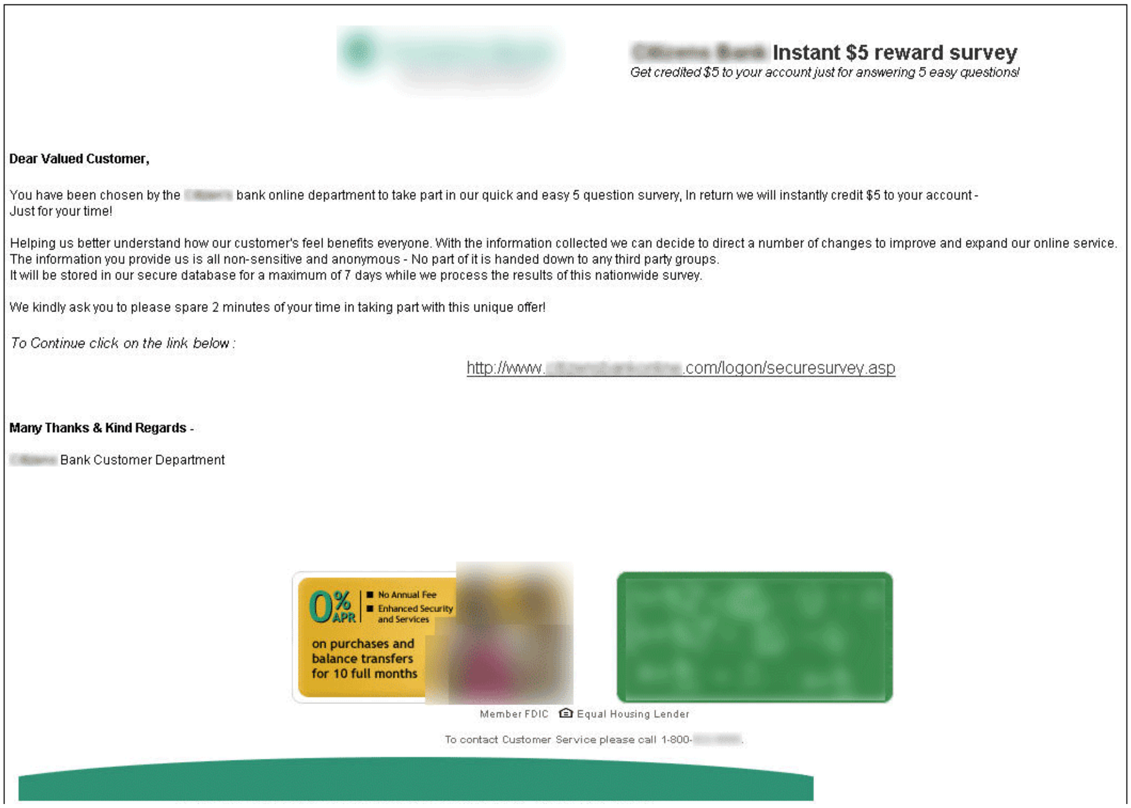
The methods used by these malware vary immensely, from basic social engineering attacks to sophisticated kernel level rootkits. Still the main functionality can be classified in a few categories, as some attacks use the same basic principles. The main divider between the methods used is the point of attack. To simplify matters we can therefore categorize the attacks into three main groups: local, remote and hybrid attacks. Local attacks happen on the victim's machine, remote attacks don't modify the machine but try to intercept or redirect the traffic of a session and hybrid attacks combine local and remote attacks and are the most powerful.

Remote attacks

Phishing

The most common remote attack against financial online services is phishing. An attacker sets up a copy of the web site they want to impersonate on a server they control. This copy includes all the code from the original site. The set of used images can be gathered during a previous legitimate session. This makes it hard to trace the imposter in the server logs as no suspicious access is made. Next, the attacker sends emails to a large number of email accounts. The emails contain a convincing message that should trick the recipient into visiting the spoofed web site and revealing his log on credentials. The example below (Figure 1) shows an attempt to lure the user to a spoofed web site by offering $5 for completing an online survey. Once the user enters his personal data into the spoofed web form the attacker saves the information and redirects the user to a fake error page or to the original web site. Many users will think they mistyped their password and do not necessarily suspect a fraud.

Phishing emails usually contain obfuscated links to the spoofed web site. There are many tricks to obfuscate the real server location, especially when HTML enabled emails are used. One such example is the method of translating the quartet of a standard IP address into a dot-less decimal number i.e. http://3639551848 [216.239.39.104]. Most browsers support these decimal IP addresses, as well as web authentication strings in the form of username:password@website.tld. So to obfuscate the URL even more the attacker can add a fake web authentication string that looks like the impersonated domain name, such as http://mySecureBank.tld@3639551848. This corresponds to a username of "mySecureBank.tld" with no password given on the decimal IP address representation of 216.239.39.104. This will trick many users into believing, that they are about to click on a link that leads to the mySecureBank.tld domain. Replacing characters with their Unicode representations or adding escape symbols to the URL makes it even harder to identify the real domain name behind a link.



[redacted] Instant $5 reward survey

Get credited $5 to your account just for answering 5 easy questions!

Dear Valued Customer,

You have been chosen by the [redacted] bank online department to take part in our quick and easy 5 question survery, In return we will instantly credit $5 to your account -
Just for your time!

Helping us better understand how our customer's feel benefits everyone. With the information collected we can decide to direct a number of changes to improve and expand our online service.
The information you provide us is all non-sensitive and anonymous - No part of it is handed down to any third party groups.
It will be stored in our secure database for a maximum of 7 days while we process the results of this nationwide survey.

We kindly ask you to please spare 2 minutes of your time in taking part with this unique offer!

To Continue click on the link below :

http://www.[redacted].com/logon/securesurvey.asp

Many Thanks & Kind Regards -

[redacted] Bank Customer Department

0% APR
No Annual Fee
Enhanced Security and Services
on purchases and balance transfers for 10 full months

Member FDIC Equal Housing Lender

To contact Customer Service please call 1-800-[redacted].

Figure 1: Phishing email from the archive of the APWG. Some details have been deliberately obscured to protect the identity of the financial institution whose logos have been used by the attacker in an attempt to reassure recipients of this email. The financial institution is not associated with the attacker or the distribution of this email, in any way.

Sometimes no obfuscation at all is used by the attacker, by simply registering a new domain name that implies connection with the spoofed name. For example:

- mySecureBank-login.tld
- my-Secure-Bank.tld
- mySecureBank-onlineservice.tld

Similar domain names can lull a user into a false sense of security.

With the introduction of international domain names (IDN) it got even worse for the user. The use of international characters in domain names introduces the problem of spoofing a domain by replacing some letters with letters from different alphabets that look the same. For example an attacker could register the domain mySecureBank.tld where the "a" is replaced with an "a" from the Cyrillic character set that looks identical. So the domain name would match the user's expectation. The attacker might find a domain authenticated SSL service, which means he proves that he owns the domain by proving that he can receive emails for it. This will let him add the convincing SSL padlock to his attempt of fooling the user. A proof of concept test of exactly this scenario was set up by Johanson [6].

Obfuscating or masquerading the destination URL are not the only tricks used by phishing emails. Once the victim is on the spoofed web site it is possible to replace the URL bar in the web browser with the impersonated domain name. JS.Trojan.Blinder [7] does exactly this, by opening a borderless white popup window containing the spoofed domain name in the right font. The position and size of this window are adapted to the user's browser so that it perfectly hides the underlying false domain name. The same method could be used to generate a false SSL padlock symbol in the browser window.

Studies by the Anti Phishing Working Group (APWG) have concluded that phishing attacks are likely to succeed with a chance of 5% on all message recipients [8]. To defend against this type of attack common sense and user education is enough in most cases. Always check the destination URL behind links in emails or even type in the known good domain yourself. To lower the technical knowledge needed by users to perform this checks many solutions have been introduced. Most browsers nowadays will warn a user if they are about to visit a web site that uses an authentication string in the URL. There are many additional tools available. SpoofGuard for example, installs a toolbar that can perform the URL checks [9]. Firstly, this toolbar checks the domain names for obfuscation with special characters, comparing against a list of known phishing domains and analyzing the code on the new web site.. The result is then presented to the user in form of a green, yellow or red status light in the toolbar. This makes it easier for a user to determine whether a web site is suspicious or not.

DNS attacks

DNS cache poisoning

Domain name checking toolbars decrease the success rate for phishing attacks described above, but other remote attacks are immune to them. In March 2005 a large-scaled DNS cache poisoning attack [10] started to fill vulnerable DNS servers with false domain name - IP address pairs. As a result, machines relying on these poisoned DNS servers received a false IP address resolution for certain domains, which led them to malicious Web sites. Detailed analyses of aspects of DNS attacks can be found in the paper from Ollmann [11]. This shows that sometimes even typing the URL address by hand in the browser might lead to a malicious site. DNS manipulation attacks are nothing new and have been around for many years. They gained media attention as attackers started to use them on a larger scale to steal logon credentials. This method of attack is sometimes referred to as pharming.

To assure that machines do not fall victim to DNS manipulation attacks, a system administrator has to make sure that his DNS servers are not susceptible to poisoning attacks. This could be done by using split-split DNS, to block recursive queries from the public or by limiting the accepted results from authoritative DNS servers as outlined by Stewart [12]. Furthermore, additional checks could be performed on the host or local resolving DNS server before looking to an external server for resolution. This could include looking at a secure database with IP addresses of the most common targeted web sites to detect if they have changed recently. Another method may be to perform a reverse lookup of the resolved IP address and check the matching domain name, however it should be noted that this is easily defeated. Nevertheless even if all these steps are carefully followed, there still is a chance that users will not connect to the legitimate site, as we will see later in the hybrid attacks section.

DNS Hijacking

In January 2005 the American ISP Panix experienced a social engineering attack [13]. Due to lax domain change verification processes, someone was able to modify the registered details of the domain panix.com. The actual DNS records were moved to a company in the United Kingdom, and Panix.com's mail was redirected to a company in Canada. Users of the domain panix.com were redirected to false sites and could have fallen victims to fraudsters if a transaction site existed at this address..

Ensuring secure interaction with domain registration organizations and proper verification from their side should eliminate these kinds of attacks.

Interception

For completeness the attack group of Interception and decryption of transmissions should be mentioned here as well. At the time of writing most online financial services use SSL/TLS to encrypt the data traffic. We assume that if up-to-date software versions are used and are properly configured on the server side, interception and modification of the session data will not be feasible for an attacker.

Local attacks

One of the biggest misconceptions is that users believe when they use an SSL connection, their online banking session is perfectly safe from a third-party. In nearly all security advice given by online services, the security experts say that if the user can see the yellow padlock symbol in the browser window the traffic between the site and the user are secured. This is true, but little does the user realize that SSL was designed to secure the channel from the user machine to the bank computer, and not the end points themselves. Whatever is done with the data before the start point and after the end point of the SSL channel is completely out of the SSL encryption context.

This fact is exploited by the PWSteal.Bankash.A Trojan [14], which captures information entered into web forms before it is encrypted by SSL and sent to the financial institution. To achieve this, the Trojan drops a DLL and registers it as a browser helper object (BHO) within Microsoft’s Internet Explorer. This threat is thus able to intercept any information that is entered in any web site visited by the browser, before it is encrypted. Loading a DLL as a BHO is not the only way to intercept entered information: injecting code into the browser's memory space or hooking common API functions achieves the same result. Displaying a spoofed web site on top of the real web site can also steal the same information for specific targets. PWSteal.Bancos.B [3] does the latter for a handful of banking web sites. When a user infected with this threat visits one of the predefined Web sites, the Trojan will generate a pop-up window, which is a copy of the real log on form. This fake window then overlays the real browser window, so that the user will enter their logon credentials into the fake form. This approach defeats the SSL encryption, as the data is never entered into the browser and therefore never encrypted. From the user's viewpoint, the opened Web site is the real bank site. The URL in the address bar is not spoofed and even the yellow SSL padlock reveals the correct certificate details, if any user should ever take the time to verify it. Only the overlaid fake password prompt is not part of the original web site and of malicious intent.

This method of attack also defeats virtual keyboards, which were introduced to protect against key logger attacks. By entering the sensitive user credentials with a virtual keyboard, displayed on a web page, the problem of interception is not solved. While a key logger is no longer capable of intercepting the secret information, multiple screenshots still can . Recording the position of all mouse clicks and one screenshot of the virtual keyboard contains enough information to recalculate the characters used. The same is true for consecutive screenshots on all mouse click events. For obvious reasons, if the information is entered into a fake pop up window then a virtual keyboard will not be able to prevent the leaking of the sensitive log on data. Regardless of how the information is entered, once it is in the browser's memory space a Trojan can steal it with the means discussed above.

Therefore a better approach to counter the previously discussed attacks is to make the gained information unusable for the attacker. In the above scenarios we were assuming that the user's credential consists of a static user name and a static password. If the attacker successfully manages to steal them, they can authenticate to the online service.

As most online services have realized, a simple static password is no longer good enough to reliably authenticate a user. Widely used solutions for this problem are time limited validation passwords. Some companies provide calculator-like hardware tokens, which generate a new one-time password every x seconds. RSA and Vasco secure tokens are good examples of such devices [15]. Other online services provide the user with a list of one-time passwords that can be used to log on. Unfortunately this does not eliminate the problem of offline password stealing attacks, it simply sets the hurdle for the attacker a bit higher. Instead of settling back and waiting while the stolen accounts pile up, they now have to use the one-time password first, before the user does. This can be achieved by intercepting the sending of the user's credentials and then blocking all Internet access of the infected machine, preventing the completion of the session. Simply rebooting the user's machine provides the attacker with enough time to use the stolen account details. However, this is easily mitigated, as some banks in Europe have demonstrated. These banks require these one time passwords not only to authenticate, but a different password for each transaction. The result is that the user would have to supply their token details repeatedly, which makes any attack more difficult, but not impossible, as we will see later.

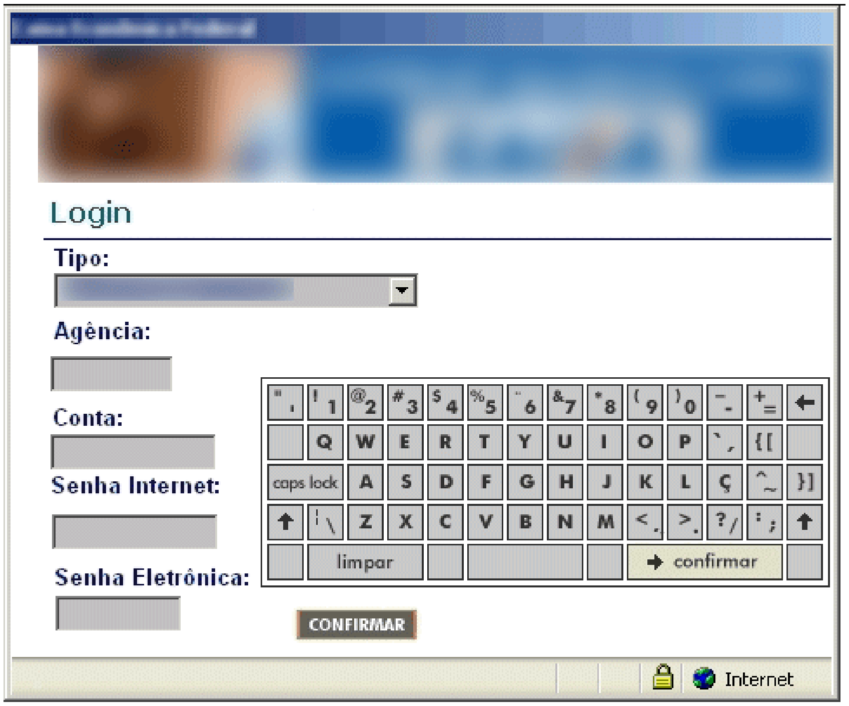


Figure 2: Fake Internet banking login prompt by PWSteal.Bancos.B [3]. Some details have been deliberately obscured to protect the identity of the financial institution whose logos have been used by the attacker in an attempt to reassure recipients of this email. The financial institution is not associated with the attacker or the distribution of this email, in any way.

The Hong Kong Monetary Authority imposed a regulation that all high-risk Internet banking applications need to provide two-factor authentication beginning in June 2005 [16]. The three common types of two-factor authentication being adopted by banks in Hong Kong are digital certificates, SMS-based one-time passwords and security-token based one-time passwords.

Another idea followed by some online banks in Europe is to lock down the capabilities of the actual user account. Any new account beneficiary has to be approved by the user. This is either done by using the same one-time password method used for authentication, which leaves some space for MITM attacks or by registering users out of band through the telephone. Unfortunately this might not be accepted by all customers as it is not very convenient. Such measures can, however, limit the damage that could be done to an online account. Some banks even let the user set a maximum amount of money that can be sent in one transaction, but once an attacker knows those limits they can stay below them.

Hybrid attacks

Nothing limits an attacker to only one type of attack. For the attacker the most successful methods are hybrid attacks that combine strategies from both local and remote attacks.

A trivial attack would be if a Trojan executed on the infected machine checked all saved bookmarks for known valuable online services and replaced the URL with a fake one, similar to phishing emails. The obvious flaw in this plan is that the user can see the modified URL if they check the address bar of the browser. So the Trojan needs to modify the browser settings to not display the address bar or overlay it with a fake pop-up window. Even though this is feasible, it resides on the same level as basic phishing attacks and can be equally well done by remote attacks.

The more sophisticated approach of the attacker would rather be to use all the power they have on the infected machine, and altering the hosts file is an obvious place to start. The hosts file gives the attacker the possibility to redirect certain domains to predefined IP addresses. This technique is used by the Trojan.Qhosts [17] in the wild. Adding the following line to the hosts file will redirect all traffic to the domain mySecureBank.tld to the specified IP address, which is under the attacker's control:

```
192.168.0.23        mySecureBank.tld
```

It doesn't matter if bookmarks are used or if the domain name is entered by hand, the user will end up at the malicious Web site. You may believe that that the hosts file is easy enough to protect by making it read only, but this scenario may be modified to include a rootkit Trojan that redirects the traffic, a DNS cache poisoning attack or a patched wininet.dll file to redirect all HTTP requests, as done by Trojan.DesktopHijack [18]. The same Trojan that alters the traffic flow can then install an unofficial root certificate authority from the attacker. Once done, the attacker can then issue their own valid SSL certificates for their spoofed web site. If a user checks the information of the certificate he will see the correct name and address details, the only thing that differs from the original is the certificate fingerprint and the IP address. This should not be a problem for the attacker, as the chance of a user knowing all legitimate fingerprints of the corresponding certificates for all his online services is very small. Symantec is currently unaware of an online service that publishes this information on their Web site, so that an interested user could verify it. Taking into account that service providers can provide this information, it still would not make a difference as all the traffic to this site is routed through our attacker. The attacker can create a spoofed certification verification page and provide the inquisitive user with the correct information on the rogue certificate. Therefore only providing this information on a secure channel, for example on paper, would solve this dilemma. As simple as this solution is, it would prevent the majority of today's attacks. Of course the attacker could generate a fake certification information window and overlay it, so that the user sees no difference even to the paper printouts.

Let us recapitulate our worst-case scenario so far. We have a machine that has been infected by some type of malware. This is not an absurd assumption, considering the propagation rates of malware today [19]. The malware is modifying the traffic flow for certain Web sites and installing a rogue root certificate; this is technically possible today, as demonstrated by some security risks such as Spyware.Marketscore [20]. Assuming temporary one-time passwords are being deployed, then using the stolen user credentials to log on can be automated by a simple script. As the Trojan is redirecting all traffic in this scenario, no online session will ever be established and the one-time password cannot be utilized by the user.

This scenario should make it obvious that we either need a reliable way of preventing traffic redirection or a more secure authentication scheme. Strong authentication of both parties to mutually authenticate each other is the desired goal. The security of the online service should not be solely based on the security of the end point (the user's PC), as a user might not be able to fully protect it, for example when the machine is shared with other persons at an Internet café.

Many ideas have evolved to solve this issue, such as anti-phishing toolbars (vide infra). Unfortunately most of these solutions only address remote attacks and contain still risks of an attacker being able to falsely authenticate himself using some of the hybrid attacks discussed above. They do increase the level of security and might work well against the phishing attacks seen today, but as the protection mechanisms evolve so does the attacker's. Of course, implementing these protection methods is better than doing nothing and raises the bar for the attacker. In the following section we will discuss some of the introduced methods and analyze their level of protection against sophisticated hybrid attacks. We will adapt the discussed worst-case scenario as we encounter new attack vectors.

Protection methods

The following general assumptions with regard to the methods of protection outlined below. The theoretical test user possesses good computer knowledge and is clever enough to recognize rogue pop-ups in Web sites. He is skeptical enough not to enter his password in a questionable Web form. An attacker is able to execute a Trojan on the remote machine and redirect traffic to their own machine. These assumptions are not unreasonable and are made to exclude the basic attacks that could be eradicated with user training. The analysis will be made from a technical perspective and ignores most aspects of user acceptance. The author is aware that this might influence the usability of a given solution, but the goal of the paper is to check the technical level of security provided by the different solutions.

SMS challenge code

Two-factor authentication systems have been introduced to ensure secure log on validation. One system that promises good user acceptance uses the user's registered mobile phone to receive an activation code. In this scenario the user identifies themselves to the bank with their account name. Next, the bank generates a random temporary password and sends it in a short text message (SMS) to the user's mobile phone number. The user enters this challenge code into the browser and proves thus that he has access to the correct mobile phone. This two-factor authentication works fine and is quite convenient for most users. One major advantage is that most users already have a mobile phone and therefore no extra hardware token needs to be bought, deployed, and supported.

If we modify our worst-case scenario described above somewhat, then we will see that this authentication cannot withstand a man in the middle (MITM) attack. The user is redirected to a fake web site. Once the user enters his account name into the spoofed web site, the attacker uses the received username, logs on to the real service and initiates the submission of the one-time password through SMS. The user will receive the code on his mobile phone and enter it into the fake web site still thinking it is the real web site. The attacker then uses the supplied code to authenticate him to the real service. With a bit of effort this attack will even work for transaction authentication. To defeat offline password stealing attacks some banks introduced the requirement to authenticate each transaction with a one-time password. Passwords generated just like the one used for the initial login process. For such a scenario the attacker will keep the fake session established and will wait till the user initiates a transaction. The attacker acts as a proxy between the real online bank and the end user, filtering out unwanted information or warnings. The attack will still work well if the user is required to reply with an authentication code through SMS instead of entering it into a web site, as the user believes that he is authenticating his own transaction.

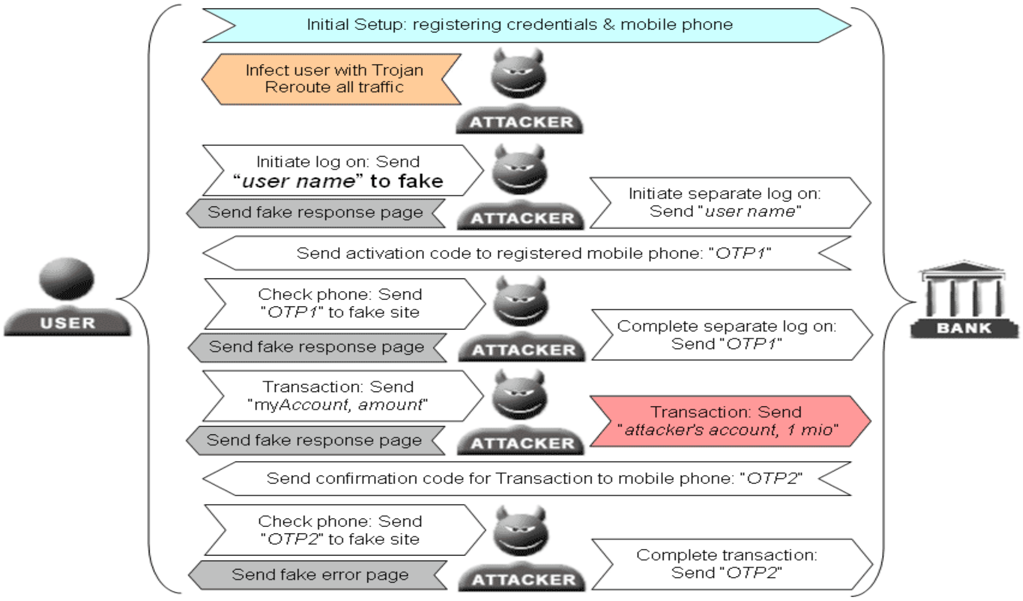

Initial Setup: registering credentials & mobile phone
Infect user with Trojan Reroute all traffic
ATTACKER
Initiate log on: Send "user name" to fake
Send fake response page
ATTACKER
Initiate separate log on: Send "user name"
Send activation code to registered mobile phone: "OTP1"
USER
Check phone: Send "OTP1" to fake site
Send fake response page
ATTACKER
Complete separate log on: Send "OTP1"
BANK
Transaction: Send "myAccount, amount"
Send fake response page
ATTACKER
Transaction: Send "attacker's account, 1 mio"
Send confirmation code for Transaction to mobile phone: "OTP2"
Check phone: Send "OTP2" to fake site
Send fake error page
ATTACKER
Complete transaction: Send "OTP2"

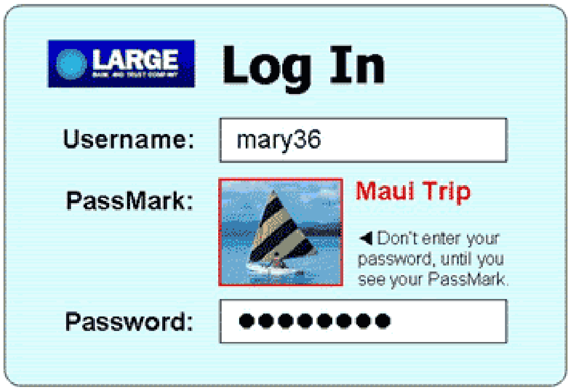
Figure 3: MITM attack scenario against SMS authentication

Authenticating each transaction would work well if it includes in the text message information on how much is transferred and to which specific account. In this case, the user would then see that his chosen target account has been replaced with the attacker's. Unfortunately this would lead to the transmission of highly confidential information via clear SMS, which might be a problem with data protection acts in place by most countries.

The same attacks and problems apply to other one-time password system like RSA secure tokens, scratch cards or iTANs.

Image verification

The PassMark system was introduced by the Bank of America in 2005 [21]. The system is based on a shared secret between the bank and the user, consisting of an image and a verification phrase. When a user wants to log on to a PassMark enabled web service, he will be prompted for his username. If the user has already authenticated to the service a "Device ID" is sent along with the username. The Device ID is realized through an encrypted cookie that is stored on the user's machine. The service then determines if the Device ID and username match and if so, present the user the login page with the secret image and verification phrase embedded in it. Users are instructed to only login if they see their known picture with their chosen verification phrase.



Username: mary36

PassMark: Maui Trip

◀ Don't enter your password, until you see your PassMark.

Password: ●●●●●●●●

Figure 4: PassMark verification image

Using our worst-case scenario with the redirected traffic, an attacker could outwit this system in the following way. As the infected machine sends traffic to the supposedly real domain it will also send all the cookies associated with this domain. Therefore no out of band traffic is needed to retrieve it. Still having full control over the target machine, the Trojan could read out all cookies and send them back to the attacker's server. The real online banking server should not rely on the client IP address for any authentication as it might change or be modified by intermediate systems such as proxy servers. Therefore, having the cookie with the Device ID and the username will be enough to retrieve the shared secret, even though the attacker can't decrypt the secure cookie. After receiving the secret information the attacker feeds the image and verification phrase back to the fake session with the user. Feeling secure by seeing their picture, the user will most likely enter his password into the spoofed web site opening the service to the attacker. As a side note it should be mentioned that it does not matter if the image is selected from a group on the online banking server or if the custom image rests on the user machine and only the path to it is stored on the main server. The attacker will mimic the real online service and provide whatever data the user expects.

An improvement made by PassMark is the replacement of the secure cookie with hardware based security tokens. This could eventually defeat the man in the middle attack, depending on the implementation. In our scenario the Trojan on the infected system could perhaps read out the hardware token, or pass the challenge received by the attacker to it, to get a corresponding answer.

In reality another security flaw can arise. If the user has no stored Device ID cookie on his machine, he will be challenged with alternative methods of authentication. These include sending a password through email or answering a predefined question. An attacker could spoof this re-authentication page. Should the user not wonder why he has to re-authenticate his machine, he will be faced with the same security problems as in the SMS challenge code approach.

Passively waiting for a legitimate session to complete and then steal the image and verification phrase for a replay attack with a local Trojan is also a feasible approach.

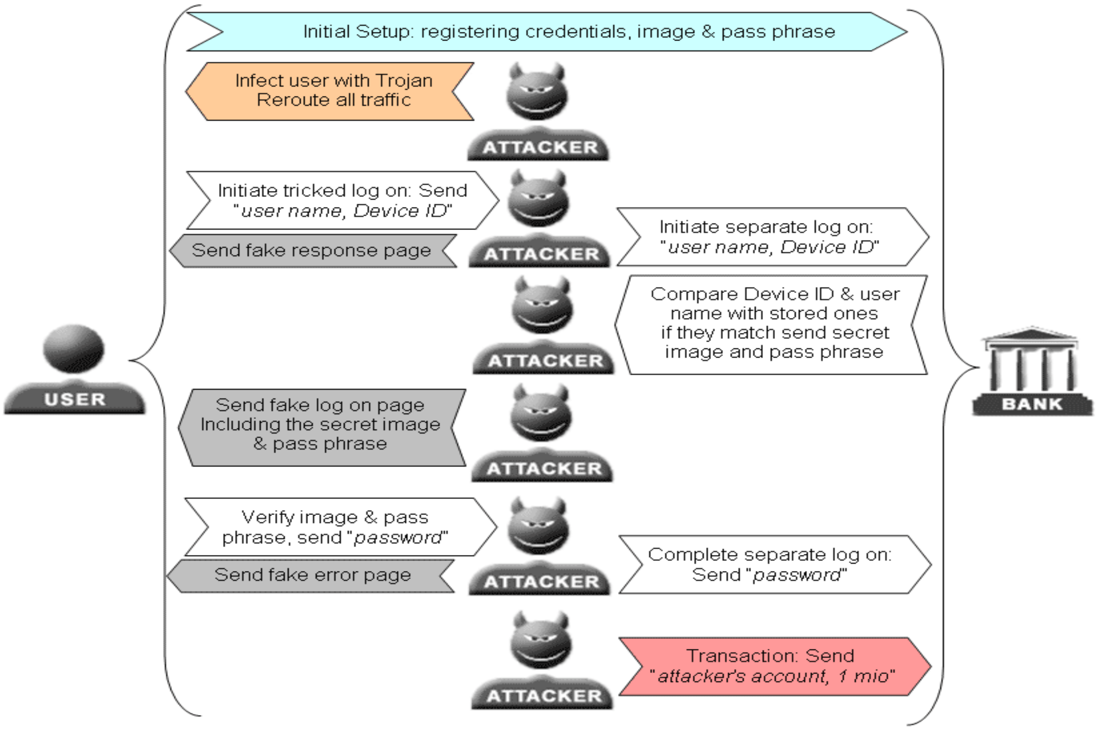

Initial Setup: registering credentials, image & pass phrase
Infect user with Trojan Reroute all traffic
ATTACKER
Initiate tricked log on: Send "user name, Device ID"
Send fake response page
Initiate separate log on: "user name, Device ID"
Compare Device ID & user name with stored ones if they match send secret image and pass phrase
USER
BANK
Send fake log on page Including the secret image & pass phrase
Verify image & pass phrase, send "password"
Send fake error page
Complete separate log on: Send "password"
Transaction: Send "attacker's account, 1 mio"

Figure 5: MITM attack scenario against image verification

Dynamic Security Skins (DSS)

Extending the image verification approach, dynamic security skins (DSS) as introduced by R. Dhamija and J.D. Tygar provide a trusted password window [22]. A photographic image chosen by the user is transparently overlaid on web forms that include sensitive information prompts. In addition a “visual hash” which can be seen as a unique graphical pattern, is overlaid as well. The visual hash is tied to the secure session and changes with each session. This makes it infeasible for an attacker to spoof a pop-up that is identical to the password prompt. But it does not authenticate both parties reliably to each other. Thus our worst-case scenario can trick the user into entering his login credentials in a MITM attack. For the DSS the whole spoofed web site is contained in one legitimate SSL session and the attack would be the same as for image verification protection.

To prevent this man in the middle attack, DSS implements the Secure Remote Password protocol (SRP) [23]. SRP is an Asymmetrical Key Exchange protocol (AKE) and provides authentication of both parties. This successfully eliminates man in the middle attacks against it. Still in our scenario we are no ordinary man in the middle attacker, we also control the client side with a Trojan. Depending on the implementation of SRP used, it might be possible for the installed Trojan to steal the necessary data from the client machine and transfer it to the attacker. The attacker can then use the gathered information to either impersonate the server in a session with the user or authenticate against the server as a client.

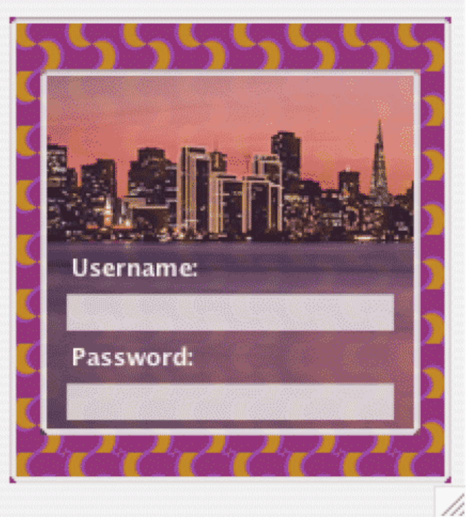


Figure 6: Dynamic Security Skins example prompt

Counter phishing methods

As seen in earlier sections, methods like SpoofGuard [9] or SpoofStick [24] work well in warning the end user of spoofed web sites when deceptive domain names are used. Unfortunately they are powerless against our worst-case scenario of redirecting traffic to a rogue site. For the fooled host (and therefore also for the anti-phishing tools installed) it does look like the communication goes to the real domain name. Even if the tool performs a reverse domain name lookup on the received IP address and matches it against an internal cryptographically signed database of correct pairs, the identity of the remote server cannot be trusted. Our example of a rootkit enabled Trojan can simply change the results received by the IP lookup in the network stack. Depending on the implementation of the local database it might also be possible to swap it and its corresponding cryptographic keys, offering the attacker the option to provide a self signed database with false information. Having a Trojan with the power of manipulating network traffic and tampering with all local files and memory regions leaves no reliable way for the tool to determine the trust state of the visited web site. Of course most attackers would simply terminate the toolbar and install a fake one that always signals a green status, instead of reverse engineer the cryptographic methods used.

Domain hashed passwords

In the same category as anti-phishing tools falls the idea of a domain-hashed password [25]. Passwords entered on a web form starting with the specific prefix “@@” will automatically be transformed into a more secure password by a browser extension. Using a one-way hash function the user-supplied password and the domain name of the web form are merged into a password hash, which is used as the new password. With this method spoofed web sites on malicious domains do not get the real password and are unable to compute the corresponding password for the real service. But as our worst-case scenario uses the real service domain name by redirecting its traffic, the password will be the real one that can be used by the attacker to authenticate themselves to the original online system.

PKI based software solution

With extensive use of cryptography and a well designed PKI it would be possible to not only authenticate the server to the user but also vice versa. This mutual authentication eliminates MITM attacks. Client certificates can provide this authentication. Secure distribution of the client's certificates and managing them on a large scale can become rather difficult. WiKID is an open source software-based two-factor authentication solution using asymmetric encryption [26]. A WiKID software client initially generates a pair of private-/public keys and exchanges the public keys with the main server. This exchange is validated using a secure channel like the telephone. Next, a secret PIN is chosen by the user and encrypted using the server's public key for transmission and storage on the server. Whenever the user needs to authenticate to a service, the WiKID client will prompt for the user's PIN. Together with the user ID this PIN will be encrypted and sent using the server's public key, making sure that only the server can decrypt and read the secret PIN. The server can then verify if the provided PIN matches the stored one and if so generate a temporary one-time passcode. This passcode is then in turn encrypted using the client's public key and sent to the requester.

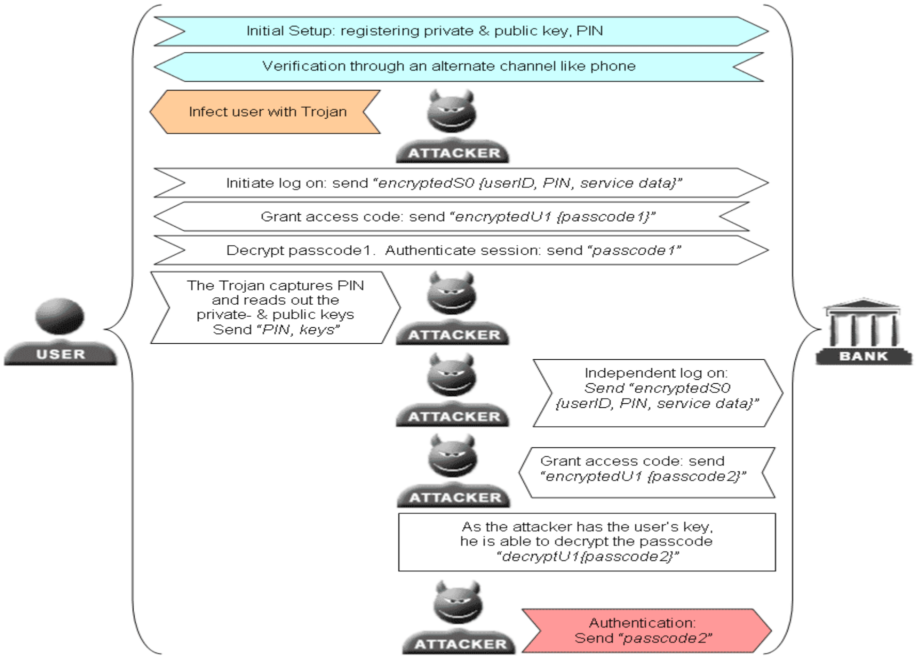

Initial Setup: registering private & public key, PIN
Verification through an alternate channel like phone
Infect user with Trojan
ATTACKER
Initiate log on: send "encryptedS0 {userID, PIN, service data}"
Grant access code: send "encryptedU1 {passcode1}"
Decrypt passcode1. Authenticate session: send "passcode1"
The Trojan captures PIN and reads out the private- & public keys Send "PIN, keys"
ATTACKER
USER
BANK
ATTACKER
Independent log on: Send "encryptedS0 {userID, PIN, service data}"
ATTACKER
Grant access code: send "encryptedU1 {passcode2}"
As the attacker has the user's key, he is able to decrypt the passcode "decryptU1{passcode2}"
ATTACKER
Authentication: Send "passcode2"

Figure 7: Attack scenario against software based PKI authentication

As only the client has the private key needed to decrypt the message, it ensures that any interceptor cannot make use of the password. As the entire solution is software based it has the big advantage of being low in distribution cost and flexible for the user as no extra hardware is needed.

Storing the client's certificate or in this case the client’s key pair in a password-protected file on the user’s hard drive, also called soft-token, leaves the problem of offline password stealing. A Trojan can intercept the PIN for the client key vault and then transmit the PIN together with the certificate or key pair back to the attacker. Once the attacker has the private key and the PIN code needed for requesting an authentication code, he can request as many passwords as he needs. Unfortunately it will never be possible to guarantee a Trojan free client by using software solutions only, as anti-malware software is always one step behind newly created malware. One possible solution to eliminate compromised clients would be to use a bootable live CD-ROM, like the Knoppix Linux distribution [27]. Booting up from a known clean read-only version of an operating system eliminates the possibility for Trojans to run during this session, unless malware manages to exploit a vulnerability on the client system during this session. A preinstalled software system like the WiKID solution can then be used to authenticate the user. Unfortunately this brings the downside that a user cannot interact with his normal system during his online session and that the user needs to bring along this bootable CD-ROM, which then becomes a kind of a hardware token. Still this is a secure and cost effective alternative to hardware based PKI solutions.

A similar solution is offered by Sentry Bay [28]. A mini CD-ROM containing a unique ID and personalized software is used to establish the connection to the bank. By using anti-hook software it ensures that key loggers can’t steal the password. A customized version of the Firefox browser verifies the bank’s SSL certificate and shuts down if it does not match the hard coded answer. Still it has yet to be proven that it is not possible for malware running on kernel level to alter the memory space of such a solution, because if it is possible malware can modify transaction data without the user noticing.

PKI based hardware token
Section 5.6 shows that a Trojan can steal the private key and PIN for a PKI based software token. Therefore tamper resistant key storage must be used to ensure high security. Smartcards with external smartcard reader devices are the most obvious solution for this. Hiltgen et al. proposed a two-stage, smartcard PKI based implementation of such a solution [29]. Pre-generated key pairs and certificates are stored on a tamper proof smartcard. Using a PIN code on the external device’s keypad unlocks the key vault in the smartcard. Therefore a key logger cannot intercept the PIN code. A signed Java applet downloaded from the bank’s web site communicates with the card reader on one side and with the bank on the other. This applet authenticates itself against the card reader. Next, it can initiate a mutual authenticated SSL channel with the bank server, signing the session. This eliminates man in the middle attacks against this schema. As the smart card securely stores the private key, it can ensure a proper authentication and prevent impersonation attacks.

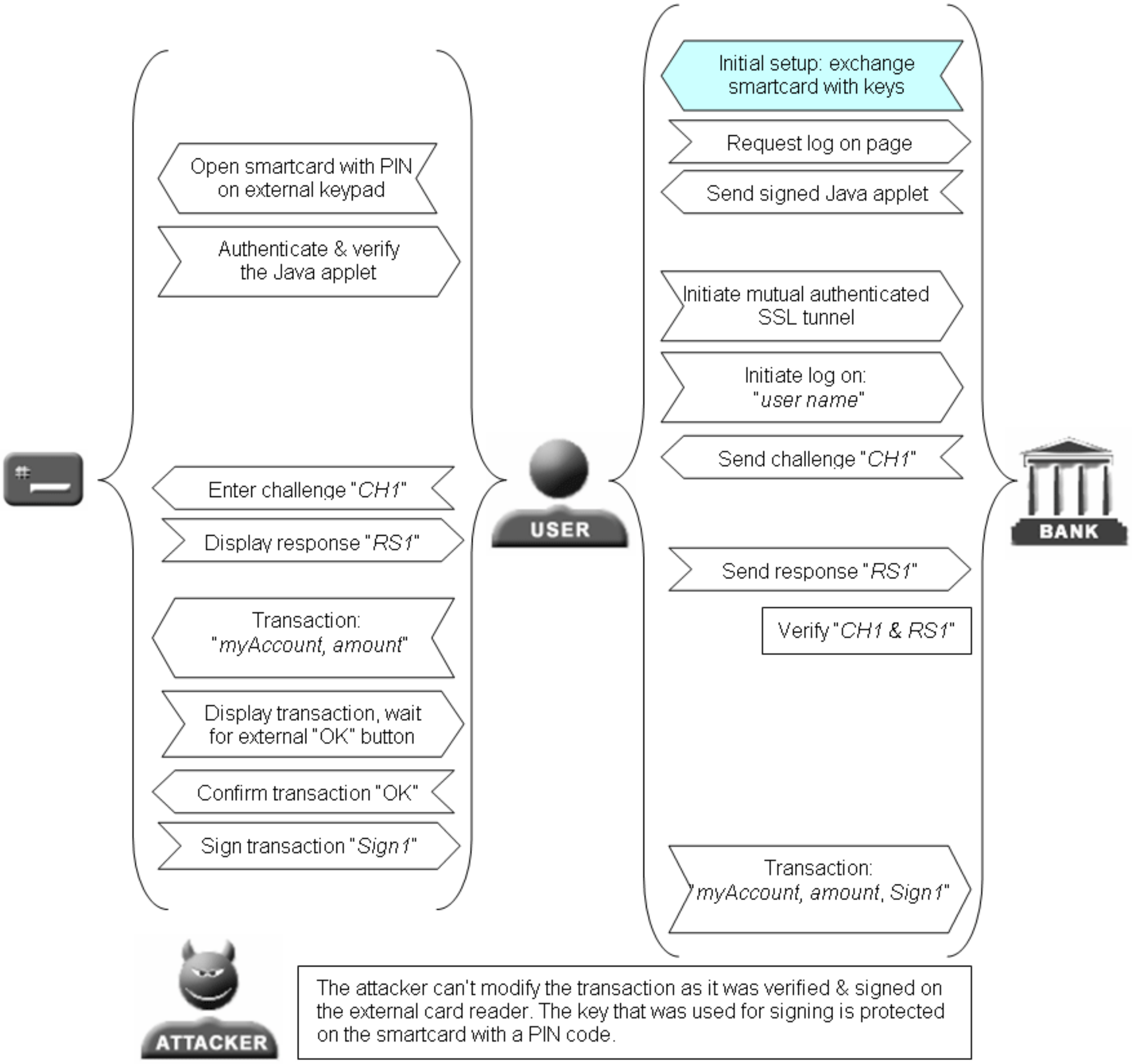

Initial setup: exchange smartcard with keys
Request log on page
Send signed Java applet
Open smartcard with PIN on external keypad
Authenticate & verify the Java applet
Initiate mutual authenticated SSL tunnel
Initiate log on: "user name"
Send challenge "CH1"
Enter challenge "CH1"
Display response "RS1"
USER
BANK
Send response "RS1"
Verify "CH1 & RS1"
Transaction: "myAccount, amount"
Display transaction, wait for external "OK" button
Confirm transaction "OK"
Sign transaction "Sign1"
Transaction: "myAccount, amount, Sign1"
ATTACKER
The attacker can't modify the transaction as it was verified & signed on the external card reader. The key that was used for signing is protected on the smartcard with a PIN code.

Figure 8: PKI based authentication with smartcards

The only practical target left for an attacker would be to manipulate the user's browser window to reflect false information. The attacker could try to replace the beneficiary account with his own and manipulate the window redraw function to still draw the old account number. To ensure security, the details of each transaction should be displayed on the external card readers' display and then signed by the smartcard. This eliminates manipulation and replacement attacks from Trojans.

Although this authentication method requires a more expensive infrastructure than the other methods proposed, it does also provide the highest level of session security.

Comparison

The following table shows a brief comparison between the protection methods discussed in this paper.

	User acceptance	System invasion	Implementation cost	Portability (Web Café)	Protection againstremote phishing attacks	Protection against sophisticated MITM
Traditional passwords	High	Low	Low	High	No	No
SMS	Moderate	Moderate	Moderate	Moderate	Yes	Depends
Image verification	High	Low	Low	High	Yes	No
Dynamic Security Skins	High	Low	Moderate	Moderate	Yes	Depends
Counter Phishing	Moderate	Moderate	Moderate	Low	Yes	No
Domain hashed passwords	Low	Moderate	Low	Moderate	Yes	No
PKI based software solution	Moderate	Moderate	Low	Low	Yes	Depends
PKI based hardware tokens	Low	High	High	Low	Yes	Yes

Table 1: Protection method comparison matrix.

Conclusion

This paper shows that the problem of online fraud has dramatically increased, and most likely will continue to do so over the next few years. The phishing attacks that we see today might only be the tip of the iceberg of what is yet to come [30]. Attackers are adapting to evade the little phishing protection tools used by users today. More powerful attacks are possible and are starting to appear on the Internet. All attack concepts described in this paper are easy enough to implement by an attacker and have already been used by malware in the wild.

Many online services are still vulnerable to basic offline password stealing attacks. Online banks are therefore now shifting to multiple-way authentication, often implementing two-factor authentication schemas, where the password page can be personalized with images or phrases. Secure one-time password tokens are also becoming widely used. These methods will set the hurdle for an attacker higher and protect against the offline password stealing attacks of today, but will not prevent the attacks of tomorrow. More drastic changes in the authentication process have to be implemented if high security is desired.

Two major attack points can be seen in the scenarios described in this paper. The first problem is contaminated client systems. Even with today’s anti-malware solutions in place, it is not possible to guarantee a non-infected client system. As statistics from virus reports in the wild show, many people still get infected with malware [30]. The controversial trusted computing model could solve the malware problem but does by now not find much user acceptance. Another promising idea is client-side behavior blocking. By blocking process manipulation and system wide hooking, tools like ProcessGuard are able to protect against key loggers and remote threat injection [31]. As with other protection tools it has to be assured that malware cannot modify the tool itself. If this isn’t ensured then the tool does not add security. Having malicious code running on a system renders most software-based solution on that system powerless. Regardless of the authentication method used, a Trojan can replace the beneficiary account number with the attacker’s, without the user or bank noticing it. Digitally signing the transaction can only work if the Trojan cannot sign the data itself. Thus can only be guaranteed by external non-infectable signing and verification methods or by using a trusted computing platform.

The second problem that needs to be addressed is a proper mutual authentication. Cryptographic authentication protocols like the Secure Remote Password protocol (SRP) [23] or the already used SSL/TLS protocol implement possibilities to solve most of the authentication problems faced today. The implementation of such a protocol must be tamper resistant, as it will operate in a possibly hostile environment. If high security is requested, private keys and certificates should be stored on tamper prove systems. Using external devices like smartcards or USB tokens for secure key storage is a possible solution for this. The Homebanking Computer Interface standard (HBCI) used in Germany shows an implementation of smartcards used with external readers that is already in use for years [32].

Unfortunately these systems are usually more expensive to roll out then simpler software solutions and might not be as convenient for users as normal passwords are. But if an online service wants to provide high security for their clients, then a sophisticated in depth approach is needed. At the risk of sounding sensational, the probability of the worst-case scenario described above occurring today might be small, but it is not zero and it is definitely increasing fast.

References

[1] Joakim von Braun, Internal study Symantec Sweden, 2005.

[2] Chosunilbo, “Breaching Online Banking Security Proves Easy as Pie”, 5 June 2005, http://english.chosun.com/w21data/html/news/200506/200506050007.html.

[3] For detailed information on PWSteal.Bancos.B, see http://securityresponse.symantec.com/avcenter/venc/data/pwsteal.bancos.b.html.

[4] For detailed information on PWSteal.Bancos.T, see http://securityresponse.symantec.com/avcenter/venc/data/pwsteal.bancos.t.html.

[5] For detailed information on Trojan.Goldun.B, see http://securityresponse.symantec.com/avcenter/venc/data/trojan.goldun.b.html.

[6] Eric Johanson, “The state of homograph attacks “, 2005, http://www.shmoo.com/idn/homograph.txt.

[7] For detailed information on JS.Trojan.Blinder, see http://securityresponse.symantec.com/avcenter/venc/data/pf/js.trojan.blinder.html.

[8] Anti-Phishing Working Group (APWG), 2005 http://www.antiphishing.org/.

[9] D. Boneh, J. Mitchell, R. Ledesma, N. Chou, Y. Teraguchi, “SpoofGuard”, 2005, http://crypto.stanford.edu/SpoofGuard/.

[10] K. Haugsness, “March 2005 DNS Poisoning Summary”, March 2005, http://isc.sans.org/presentations/dnspoisoning.php.

[11] G. Ollmann, “The Pharming Guide” , 2005, http://www.ngssoftware.com/papers/ThePharmingGuide.pdf.

[12] J. Stewart, “DNS Cache Poisoning – The Next Generation”, http://www.lurhq.com/dnscache.pdf

[13] T. Cole, B. Tonkin, “Email from Tim Cole to Bruce Tonkin [regarding the unauthorized transfer of panix.com domain]”, 14. March 2005, http://www.icann.org/correspondence/cole-to-tonkin 14mar05.htm.

[14] For detailed information on PWSteal.Bankash.A, see http://securityresponse.symantec.com/avcenter/venc/data/pwsteal.bankash.a.html.

[15] RSA Security, “RSA SecurID Authentication”, 2005, http://www.rsasecurity.com/node.asp?id=1156.

[16] Hong Kong Monetary Authority, “Launch of Two-factor Authentication for Internet Banking”, 30 May 2005, http://www.info.gov.hk/hkma/eng/press/2005/20050530e3_index.htm.

[17] For detailed information on Trojan.Qhosts, see http://securityresponse.symantec.com/avcenter/venc/data/trojan.qhosts.html.

[18] For more information on Trojan.DesktopHijack, see http://securityresponse.symantec.com/avcenter/venc/data/trojan.desktophijack.html.

[19] Symantec, “Symantec Internet Security Threat Report VIII”, September 2005

[20] For more information on Spyware.Marketscore, see http://securityresponse.symantec.com/avcenter/venc/data/spyware.marketscore.html.

[21] B. Riess, “Bank of Amercia announcement”, 26. May 2005 http://www.passmarksecurity.com/BofA.jsp.

[22] R. Dhamija, J.D. Tygar, “Phish and HIPs: Human Interactive Proofs to Detect Phishing Attacks”.

[23] T.Wu, “The Stanford SRP Authentication Project”, 17. August 2005, http://srp.stanford.edu.
[24] SpoofStick, http://www.spoofstick.com.
[25] D. Boneh, C. Jackson, J. Mitchell, N. Miyake, B. Ross, “Web Password Hashing”, 2005 http://crypto.stanford.edu/PwdHash.
[26] WiKID Systems, “The WiKID Strong Authentication System”, 2005, http://www.wikidsystems.com.
[27] Knoppix, “From zero to Linux in 5 minutes”, 2005, http://www.knoppix.net.
[28] SentryBay, “Online Bank Card”, September 2005, http://www.viralock.com/onlinebankcard/index.htm.
[29] A. Hiltgen, T. Kramp, T. Weigold, UBS AG & IBM Ruschlikon, "Secure Internet Banking Authentication", 15 March 2005, http://www.ubs.com/1/e/ubs_ch/authentication.html.
[30] Symantec DeepSight Research Report,” Online Fraud Communities and Tools", 24/01/2006
[31] Diamond CS, “ProcessGuard“, September 2005, http://www.diamondcs.com.au/processguard.
[32] Bundesverband Deutscher Banken, “Homebanking Computer Interface”, 10.5.2000.

About the author

Candid Wüest graduated in computer science at the Swiss Federal Institute of Technology (ETH). He extended his experience in IT security during the last ten years while he worked for several companies, including the global security analyzing laboratory at IBM Research, Rüschlikon. He has been with Symantec for the last two years and is currently working as a virus analyst in Symantec Ltd. Dublin. He has published various papers and articles in magazines, given interviews to radio and newspapers, and presented at many conferences, such as COMDEX Scandinavia.

About Symantec
Symantec is the global leader in information security, providing a broad range of software, appliances, and services designed to help individuals, small and mid-sized businesses, and large enterprises secure and manage their IT infrastructure. Symantec's Norton™ brand of products is the worldwide leader in consumer security and problem-solving solutions. Headquartered in Cupertino, California, Symantec has operations in 35 countries. More information is available at www.symantec.com.

Symantec has worldwide operations in 35 countries. For specific country offices and contact numbers, please visit our Web site. For product information in the U.S., call toll-free 1 800 745 6054.

Symantec Corporation
World Headquarters
20330 Stevens Creek Boulevard
Cupertino, CA 95014 USA
408 517 8000
800 721 3934
www.symantec.com