CURRENT ADVANCES IN BANKING TROJANS

Candid Wüest
Symantec, Andreasstr. 15, 8050 Zürich, Switzerland

Email candid_wueest@symantec.com

ABSTRACT

For ten years we have been fighting against malware that targets online banking. Trojans like Zeus, SpyEye, Torpig, Carberp and others still manage to loot millions of dollars from infected user accounts every year.

This paper will analyse the current situation of online banking malware. How sophisticated are the current versions of these trojans and how did they evolve? Which techniques are currently used to bypass the security measures of online banking applications? Are man-in-the-browser attacks still the most sophisticated ones? Are other attacks like proxies or DNS redirections taking over? How much do the attackers focus on mobile banking or tokens on mobile phones like mTANs? These mobile features have been introduced to create a second authentication channel, independent from the infected PC, in order to protect against trojan attacks and are therefore of interest to the attackers. We will dissect new features like the P2P option of Zeus but also lesser known methods like the *Firefox* XUL injection used by Trojan.Neloweg.

DEFINITION OF BANKING TROJANS

Of course there are many generic keyloggers that are able to capture log-in credentials, and depending on the implemented security features of the service they might be able to successfully misuse this information as well. For the scope of this paper we will only focus on dedicated trojans that specifically target at least one financial institution and are capable of evading more advanced protection schemes that use more than just a static username/password combo for authentication. We assume that the malware infects at least one of the user's local systems. It should be obvious that many of the discussed malware samples could be modified to steal credentials for other services like social networks, gaming sites, and free mailers.

COMMON FEATURES

Below is a list of some of the most common features of today's banking trojans. They might not be present in all specialized banking trojans, but they reflect the state of evolution of this threat class.

Botnet structure

A common feature of banking trojans, as with any trojan, is the botnet communication structure with a command and control (C&C) server. Each infection will report back to the C&C, usually a web front-end consisting of a few PHP scripts. The attacker can then send commands to all or individual machines. These commands usually relate to updating the installed version of the trojan or initiating fraudulent transactions. An exposed C&C server is also one of the weak points that could be shut down by law enforcement, hence some trojans, such as Zeus, have started to explore P2P communication as well.

Web injects

Modern banking trojans often need to interact with the browser since this is the interface for common online banking platforms. One of the easiest ways to achieve this is to employ a man-in-the-browser (MITB) component. This is not a new concept and we wrote about it back in 2005 [1]. It means that the trojan will inject some code into the running browser process or hook networking APIs in order to monitor and modify incoming and outgoing data. This is done before any transport layer encryption SSL/TLS is applied to the traffic. Older variants used BHOs, LSPs or enumerated window titles in order to grab passwords.

If inline hooking is done, then typically the following APIs are hooked:

- wininet.dll : HttpSendRequestA/W
- wininet.dll : HttpSendRequestExA/ExW
- wininet.dll : InternetReadFile
- wininet.dll : InternetWriteFile
- wininet.dll : InternetReadFileExA/ExW
- nspr4.dll : PR_Read
- nspr4.dll : PR_Write
- WS2_32.DLL : send

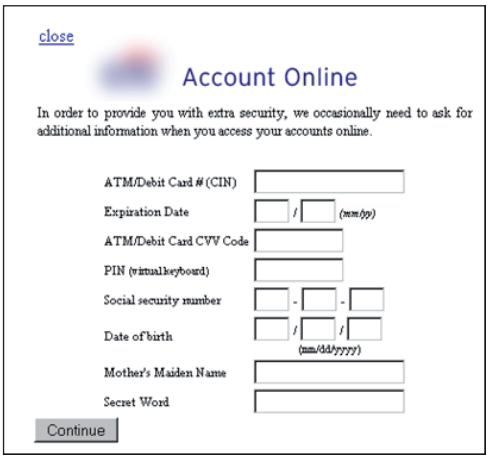

Figure 1: Injected web form asking for additional information.

Some trojans, like Zeus, have developed their own scripting language for web injects. This allows the attacker to easily adapt to new banks. A simple script will search for given patterns in the HTML code when the URL matches a given regular expression. Once found, a new HTML or script code will be injected at this position before it is rendered. This gives the attacker full control over the representation layer without breaking any of the given security features such as security seals or SSL cert padlocks. For the victim, the injected code appears to

be trusted code from the bank. This is frequently done to insert extra fields for credit card numbers, adjust the account balance after money has been stolen, or modify transaction details so the user approves it. This simple method is very powerful and can break most security measures.

Like the trojan itself, these web injects are traded on underground forums and can even be ordered for specific banks if they are not yet readily available.

```
Webinjects sale Zeus/Spyeye
THIS SERVICE WILL SOON GO IN PRIVATE MODE


Hello

We sell already made webinjects for Zeus/Spyeye. We can
develope webinjects to your needs if you provide logins for
testing it. Injects can be made on for any country and any
language if you provide details for it.
All injects are tested on accounts before selling. We can
do injects in different languages, depending on your needs
(you have to provide the text for fields)
Injects are sold encrypted and you can't modify them.

[..snip..]

Price for one inject is now 60 WMZ/LR

Price for UK injects pack 800 WMZ/LR
Price for US injects pack 740 WMZ/LR

Updated/modify of injects 20 lr each.
```

Figure 2: Custom web injects for sale.

Some banks do offer standalone banking applications, especially for smaller enterprise customers. Due to the smaller user base and the added complexity of these dedicated applications they are not yet in the focus of the common malware industry – but this does not mean that they would be immune to such attacks.

Traffic redirection

There are two different concepts used for traffic redirection. One is to redirect infection traffic from victims that the attacker does not want because they fall out of the targeted scope. This often happens before they actually get infected. Typically, this is done at the level of the infecting website of the drive-by download toolkit. The user's IP address is checked against a list of accepted regions and if it does not match any desired country it will be forwarded off. Often, traffic distribution systems (TDSs) are used to resell the traffic. The new destination could then be another cybercriminal who will attempt to install something else on these machines or a classic advertisement program that generates revenue.

The second principle, known as pharming, is to redirect all traffic, or at least specific domains, at the infected machine. Two simple ways to achieve this are by modifying the hosts file or by changing the user's DNS server to one under the attacker's control. This can be used in combination with the installation of a rogue digital root certificate, allowing the attacker to fake SSL connections for any domain. This allows the attacker to perform a man-in-the-middle (MITM) attack against interesting connections. For example, they can redirect the connection of an online banking session through their own server. Since they control the local certificate the victim will not notice anything unless he knows how to check the fingerprint of the certificate.

For the bank it is a normal session initiated from the attacker's machine, which will transparently pass all the authentication challenges through to the victim. This is one of the weak points of this attack because the bank will notice that the session was initiated from an unusual IP address which might flag internal anti-fraud mechanisms. It is also more complicated for the attacker compared to local injections and not so common any more. We do not think that this will replace local MITB attacks in the future.

For all non-financial sites, the attackers can still earn money by replacing existing advertisements or even adding their own banners to every site the user visits. This will result in a massive click-fraud that can pay out well for the attacker. We have seen this method becoming more popular among non-banking trojans, but often done with local injections as well.

Various common features

Depending on the targeted service, there are other common features among the banking trojans:

- Stealing locally stored client certificates of financial applications
- Keylogging for static passwords
- Dropping all browser cookies in order to force the user to retype all credentials
- Removing other malware or blocking security tools
- Stealing locally stored passwords for different services for further profit and propagation
- Creating screenshots
- Anti-debugging features to slow down analysis
- Modular updates or plug-ins.

Of course, there are many different variants floating around and some are modified to include additional features or download other components, especially since some of the sources of these toolkits have been leaked and can now be modified and adapted by anyone.

INFECTION VECTORS

The infection vector for banking trojans is similar to other malware types. Each local group has their preferred way of spreading the malware, sometimes depending on the targeted community. One of the most popular attack vectors is that of drive-by download sites. They are often created with attack toolkits like Blackhole, Nuclear Pack or Phoenix, and are sometimes sold as a bundle with the banking trojan in the underground. Once a toolkit becomes too popular, the more advanced attackers will switch to another one. The infected URLs can then be pushed by black hat SEO attacks, mass defacement SQL injections, or infected banners. Also, the spreading of malware in social networks is rising since users often blindly trust posts from friends. This allows such infected links to spread quickly through the net.

In addition to infected websites, we still see email campaigns with malware attached, often disguised as an invoice from a

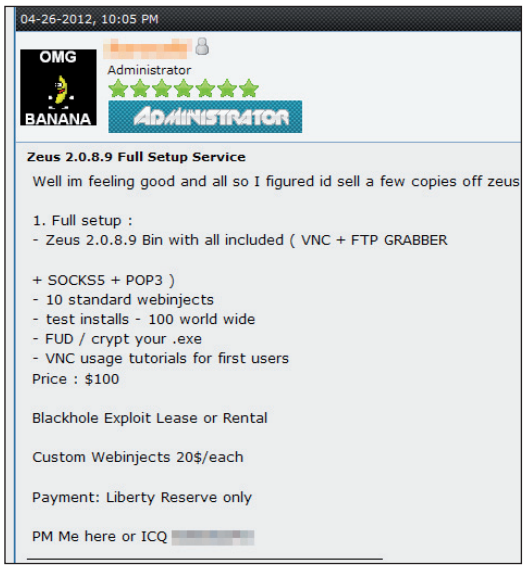

Figure 3: Zeus installation bundle.

shop or shipping company. But with many companies and email hosting providers rigorously filtering out all attachments, the method is becoming less successful.

BANKING TROJAN EXAMPLES

Some banking trojans are heavily focused on specific geographical regions and while they might not play a major role on a global scale, they can be very relevant for the local market. For example, Carberp and Sheldor are very active in Russia. There are a few dozen active banking trojans around and we have focused on those with new functionalities.

Trojan.Zbot (Zeus)

Zbot, or Zeus, is one of oldest and probably most widespread banking trojans. It comes with various plug-ins available like backconnect for sockets or Jabber notification. The original author has stopped public development, but there are a few strains which have been actively developed since the source code was shared.

One of the most noteworthy evolutions we saw last year was a Zeus variant that uses a P2P infrastructure. This might have been a reaction to multiple Zeus botnet takedowns where, in a coordinated action, multiple C&C servers of big Zeus networks were taken offline, making the botnet headless and therefore cutting off any new commands from the attacker.

In this variant, each bot has a list of tuples with the IPs of known peers and SHA1 hashes of a unique identifier for each client. Zeus will contact each of them by UDP. Once the initial authentication is exchanged the two bots can exchange configuration data and lists of other peers. Earlier versions used the P2P network to exchange URLs for C&C servers which were then contacted in the normal fashion. Newer versions use P2P exclusively and do not fall back on HTTP communications for C&C.

The encryption used by the bot for the configuration files has gone beyond the 'standard' Zeus bot strain. With the new version, the data is still compressed, XORed with the preceding byte, and encrypted with RC4, but there is now an additional layer: a byte-per-byte XOR applied to each block of the configuration, with a dynamically calculated encryption key, but still no asymmetric encryption or signing of data is used.

This change obviously makes the botnet more resilient to takedowns and makes it harder to track. Sites like the Zeus Tracker from abuse.ch currently track around 700 known C&C servers globally for Zeus [2]. Depending on the exact implementation, it is possible to create rogue clients that can enumerate peers and send false commands, since the encryption of the configuration files is known. As a side note, we have also seen versions of Zeus and SpyEye that contain fake C&C server domains, probably to confuse researchers and tracking sites. A full Zeus manual can be found online [3].

SpyEye

SpyEye is very similar to Zeus and there are even some strains which combine sources from both kits. The developer started to focus on the use of plug-ins consisting of DLLs and corresponding cfg config files, allowing the users to expand the functionality in an easy way. Active development of SpyEye seems to have stopped at the end of last year. But of course the trojan is still actively used and a few dozen different plug-ins are in the wild, from the usual keylogging and cert grabber modules to DDoS plug-ins [4].

Just like Zeus, SpyEye is capable of using the same web injects to modify the rendering of specific sites, which makes it very popular and flexible [5].

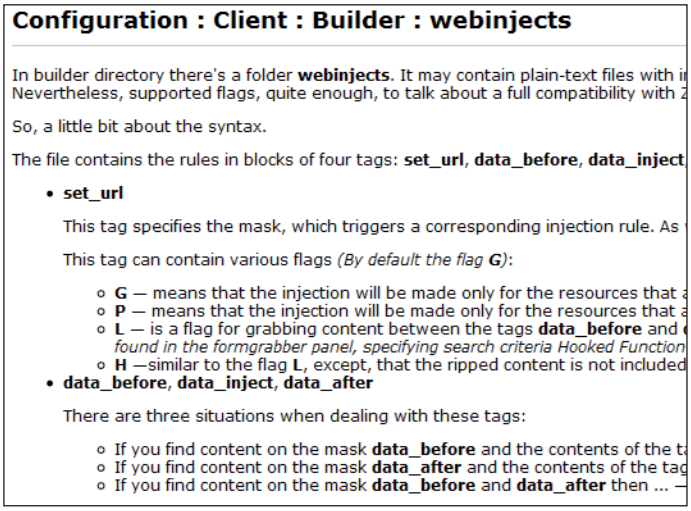

Figure 4: Web inject explanation from SpyEye manual.

Trojan.Neloweg

Trojan.Neloweg is a newer species and comes in two parts: an installer and a dropped DLL. This trojan is not that widespread, but uses some interesting, unique features. So far we have seen it mainly targeting users in the Netherlands and the UK [6].

The installer creates registry entries to configure aspects of the threat and ensure its persistence. The data is stored under HKEY_CURRENT_USER\UDP.

In order to be able to interact with the browser, Trojan.Neloweg uses the WSCInstallNameSpace API to create a new namespace and associate it with Winsock2 and its own DLL. This ensures that the trojan DLL is loaded into memory whenever a program uses Winsock2 to communicate with the Internet. An internal filter list checks into which process the DLL was loaded and terminates itself if it is not a desired process, like a browser.

The trojan works with the most common web browsers and the injected DLL will perform the malicious actions, just like other banking trojans. But if *Firefox* is used, a special routine will be started. The trojan drops its own *Firefox* extension in the following core folder:

%ProgramFiles%\Mozilla Firefox\chrome\error.manifest

%ProgramFiles%\Mozilla Firefox\chrome\error.jar

%ProgramFiles%\Mozilla Firefox\components\nsLego.js

%ProgramFiles%\Mozilla Firefox\components\nsILEgo.xpt

The trojan will then locate the current user's *Firefox* profile folder and delete the xpti.dat and compreg.dat files, responsible for the user's settings and registered components, to ensure that the new component can be inserted and loaded.

The most interesting file is error.jar, which contains the XUL files and the JavaScript files that implement the bot functionality. Since *Firefox* extensions do have access to the file system, the registry and the network, a fully functional bot can be implemented within the browser. The supported commands include:

- !cmd! – download and execute a file
- !block! – block a given list of websites
- !screen! – send back the content of a website

```
var actions=new actions();
window.addEventListener("load",function(){myExtension.init()},false)
window.addEventListener("unload",function(){myExtension.uninit()},f
window.addEventListener("load",function(){myExt.init();},false);

var wrk=Cc["@mozilla.org/windows-registry-key;1"].createInstance(Ci.
var nsIE=Cc["@mozilla.org/process/environment;1"].getService(Ci.nsIE
var nsIL=Cc["@mozilla.org/file/local;1"].createInstance(Ci.nsILocalF
var CMD_TICKIT="!tickit!";
var CMD_EXEC_FILE="!cmd!";
var CMD_BLOCK_URL="!block!";
var CMD_SCREEN_URL="!screen!";
var CMD_CONTENTPROCESSING_URL="!content!";
var CMD_REDIRECT_URL="!redeR!";
var CMD_KILLBOT="!kill!";
var CMD_GETSTORAGE="!storage!";
var CMD_FILTER_URL="!filter!";
var CMD_ALT_URL="!alt!";

var ROOT_KEY="UDP";
var gtURL="";
var INFO="";
var BID="";
var tick=30;
var VERSION="0.00";


try
{
wrk.open(wrk.ROOT_KEY_CURRENT_USER,ROOT_KEY,wrk.ACCESS_READ);
if(wrk.hasValue("g"))gtURL=wrk.readStringValue("g");
if(wrk.hasValue("i"))INFO=wrk.readStringValue("i");
if(wrk.hasValue("id"))BID=wrk.readStringValue("id");
if(wrk.hasValue(CMD_TICKIT))tick=parseInt(wrk.readStringValue(CMD_T
```

Figure 5: Bot communication code in the Neloweg Firefox extension.

- !filter! – steals form data for websites matching specific keywords
- !content! – inject code into specific websites
- !alt! – specify alternative C&C server.

We discussed *Firefox* extension malware a few years ago and highlighted that it might be missed by many security tools, but we have not yet seen it being used by a banking trojan to this extent [7]. Of course, since the extension files are stored normally on disk, anti-virus tools can detect and delete them, but they will automatically be recreated by the malicious DLL. It is interesting to note that the XUL part of the threat could easily be ported to other platforms like *Mac OSX*.

Trojan.Tatanarg (Tatanga)

This trojan, with MITB functionality, emerged in 2010 and includes all of the expected functionalities of a banking trojan. It is component-based, which means the initial installer downloads several encrypted components that perform various functions. This allows the trojan to update and exchange individual modules and make them harder to detect. The provided functions include the following:

- Modifying HTML in the browser – as with web injects
- Removing specific Zeus variants – similar to SpyEye
- Disrupting security software – relatively common in malware
- Enabling *Windows* remote access – others started using VNC or RDP for remote access
- Receiving and executing further commands.

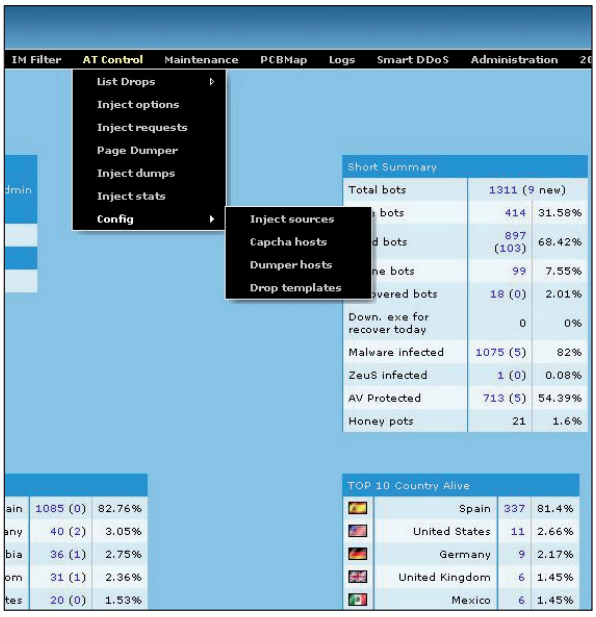

Figure 6: Tatanarg C&C panel.

Like other sophisticated banking trojans, it is able to initiate fraudulent transactions in the background. In order to adapt to changes on the online banking sites, the trojan is capable of

dumping and sending the content of the banking site to the C&C server. This allows the attacker to modify the injection code if necessary.

As is visible in the C&C panel, the threat reports back if the victim is using AV tools and if it is suspected to be a researcher's machine or a honeypot. This allows the attacker to carefully select his victims and tweak the threat for future attacks.

Interestingly enough, the trojan has an unusual method to perform local MITM attacks against SSL traffic. In contrast to other trojans which intercept and modify the traffic before it gets encrypted, it installs a local proxy and self-signed certificates. This allows the trojan to monitor and modify any traffic sent over HTTPS to banking sites [8].

IMPLEMENTED SECURITY MEASURES FROM THE BANKS

Most banks implement some internal anti-fraud detection mechanisms to detect suspicious transactions. These work in addition to the technical security measures around the authentication and transaction process. Most banks have started to assume that the user's machine is infected by malware and therefore that security features need to be built on such a premise. Unfortunately, experience has shown that this seems to be a valid assumption.

Below is a selection of the most common security measures in online banking and examples of how they can be defeated.

Virtual keyboards

Virtual keyboards were introduced a long time ago to defeat generic keyloggers. One of the first responses from the trojan authors was to record screenshots or video around the mouse pointer to capture the virtual keys. This works fine, but needs some post processing with OCR to automatically get the entered code. Most specialized trojans today inject a component into the browser or into the network stack in order to capture the data before it is encrypted by transport security like SSL. This works no matter how the credentials are entered. The method of injecting into the browser, sometimes referred to as form grabbing, is a simple but effective technique because the pass code needs to be transmitted to the bank in the end. Therefore, virtual keyboards bring little to no additional security against specialized banking trojans. This might be one of the reasons why some banks have stopped using them in newer systems.

Two-factor authentication

Two-factor authentication (2FA) is widely adopted by online banking systems. In some countries, like the USA, it is even required by regulations. Unfortunately, it is often only used for authentication but not for transaction signing and can therefore be bypassed. There are different implementation systems that can all be grouped into the same class. The 'something you know' part of 2FA is the password of the user. The 'something you have' part can be a scratch list of one-time passwords (OTPs) provided by the bank, a token that generates a new OTP every 30 seconds based on a shared secret (hardware or software tokens exist) or a mobile TAN where an OTP is sent by text message to the registered phone.

Starting with the assumption that the local machine is infected with a trojan, we can completely ignore the log-on process to the online banking system for now, since the trojan does not care. It is true that the trojan could steal the log-on credentials and send them back to the attackers, who might be able to use them in a small time window to log on. But we rarely see this done nowadays as there are easier ways, and things like restrictions on unregistered IP addresses make it harder to succeed. Hence, the local trojan simply waits until the user has successfully logged in and rides on this session. If the bank system does not require transactions to be authenticated, then the trojan can already start issuing new transactions from within the browser at this point. There are still a lot of systems that, by default, do not require any additional authentication for transactions once the logon is completed.

If the transaction needs an OTP to be entered by the user, then the trojan can simply wait for the user to make a transaction and modify the details on the fly. The trojan will act as a local MITB and change the beneficiary account details in the background. Once the bank asks for the OTP, the trojan will swap the details back, so that the user thinks the desired transaction has been issued and authorized. The swap is completely invisible to the user. Swap attacks are one of the inherent problems with OTPs, which might be bound to a transaction, but do not reveal to the user which transaction that is. Most of the time, OTPs can even be used for any transactions, making the swap even easier.

This shows that OTPs do make classical phishing more difficult, but do not pose a big hurdle for local banking trojans as of today.

Mobile TAN

Especially in Europe the so-called mobile transaction number system, or mTAN, is very popular. The bank sends an OTP to the previously registered mobile phone of the client. This can either be done for login or for authorization for each transaction. This text message usually consists of a numeric code which will be valid for 30 seconds, similar to hardware tokens.

There are various social engineering tricks that can work against mobile TANs and many other security measures. For example, we are aware of cases where Zeus used web injects to display a notification message on the welcome page after the logon. The user was told that there had been a false transaction credited to the account that needed to be reversed or else the account would be frozen. When checking the balance there seems to be more money than expected, credited after an unknown transaction. All those facts are faked and just HTML code injected by the trojan. The message then tells the user to fulfil a transaction in order to send the money back to the original account. The user will be guided to a pre-filled transaction page, and if the social engineering was convincing enough, submit the transaction.

Other cases report that Trojan.Tatanarg will tell the user to perform a test transaction. The user is persuaded that nothing will be deducted from the account and that it needs to be performed in the name of security. Although less convincing

than the above story some users might still fall for it, since it looks like the message comes from the original banking site.

The social engineering attacks obviously work against any security features including challenge response with chip cards, since the user wants to initiate the transfer. But we have also seen attacks where the trojan changed the registered mobile phone number. This will normally generate a confirmation SMS to the old phone number or email that needs to be confirmed before it is switched. With the usual social engineering stories the user is made to believe that an mTAN should be entered for an additional security check. Once the phone number is confirmed this way, the trojan can start issuing fraudulent transactions in the background. The confirmation mTANs will be sent to the new mobile phone number, which is often just a gateway to an email service and ends up at the C&C server. The trojan then receives the OTPs and completes the hidden transactions. The attacker can even switch the phone number back.

Since mobile threats have become more popular it is not surprising that about two years ago we started seeing banking trojan modules for mobile phones, the main one being a Zeus module, also known as Zeus in the mobile (ZITM), and a similar module for SpyEye. Both are available for various platforms including *Symbian*, *Windows Mobile*, *BlackBerry* and *Android*. Once installed the mobile part of the trojan can forward any text message received without the user even noticing that a message was received. The installation process relies heavily on social engineering. For example, one Zeus attack displayed a message to the user saying that there had been a lot of incidents with SIM card cloning and that a special security tool needed to be installed on the mobile phone in order to secure it.

There are a few reports where the criminals actually reported the victim's phone as stolen in order to get a cloned SIM card from the provider. This obviously works, but does not scale well and therefore will not be discussed further.

All of the above-mentioned methods work, even when the text message explicitly says where the money will be sent, as the user is either tricked or never sees the message. The mobile TAN procedure does make it more difficult for the attackers since they need to coordinate and take over two devices – but it is not impossible, and the possibility of huge profits is enough motivation for the attackers to develop new strategies.

Challenge response (Chip)

Some banks have implemented challenge response systems for authentication or for transaction signing. One of the most common methods is through the use of an external smart card reader and a smart card issued by the bank. The typical implementations will display a challenge in the browser that needs to be entered into the card reader. Together with a PIN code the information is processed by the chip on the smart card and produces the response which is displayed on the smart card reader. This needs to be entered in the browser and sent back to the bank which can verify the result. The less secure implementations issue a challenge which is not bound to the transaction, where the correct answer can be used to authorize any transaction. Some use a smart card reader that does not display the transaction details, which means that the user does not know which transaction is about to be authorized. Such implementations are prone to swap attacks and can be treated and therefore bypassed like any OTP method described above. Only if the transaction details are shown on the independent display can the user be sure to know which transaction the challenge belongs to.

In 2010 some interesting variants of Zeus were seen in Russia [9]. By monitoring for USB devices the trojan is able to detect if any USB smart card readers are connected to the PC. Once an interesting device is found the trojan waits until the user unlocks the smart card and then signals this state back to the C&C server. The attacker is now able to covertly issue its own APUD commands to the smart card, potentially making it possible to generate additional transaction codes.

Mobile banking

We have already seen a few online banking applications for mobile devices on the market. These applications allow the users to check their balance and issue transactions from within the mobile phone app [10]. Some of the earlier apps only provided public news feeds or acted as a web browser to log into the normal online banking site. As of today we are not aware of any mobile trojans that specifically target a mobile banking app, but given the early adoption stage of the users and the fact that mobile devices are often not protected through security software, this may change in the future. This starts to be especially interesting for the attacker when out-of-band (OOB) authentication is used by the bank in the form of an mTAN text message. Since the banking session and the OOB authentication is happening on the same device we are losing the second channel, and therefore weaken the overall security concept. Most vendors seem to rely on the fact that mobile apps are running in a sandbox and cannot be manipulated by other apps. This is mostly true, but of course only as long as the device is not rooted or jailbroken by the user or an exploit is used to bypass the security guards.

It should also be noted that classical phishing could regain some popularity since the space of the URL bar on mobile devices is very limited and often faded out. Therefore, opening a phishing URL such as 'login.mybank.com.hackedomain.tld' might go unnoticed by the user, because 'login.mybank.com' is all that is displayed in the URL bar.

We have also seen a few attempts to place fake banking applications on the app markets that would operate as a web browser and log the static log-on credentials. These attacks were not successful against 2FA or similar authentication methods. Newer versions of these threats pose as OTP token generators for the mobile phone. The smart idea of the attack is that once it is installed the user has to authenticate himself with the account number and the password before the fake OTP is shown. This allows the attacker to steal all the necessary log-on credentials and later, the mTAN, from the same device without having to infect the corresponding PC. This makes breaking the 2FA really simple, but it might also be easily spotted by the user since the OTP does not work. Currently, it seems that the

transaction will be initiated from the C&C server, but it is plausible that the attackers will implement a back connection module and issue the transaction directly from the mobile phone in order to bypass suspicious IP filtering [11].

Log-off refusal attack

An interesting approach to bypass some of the security features was implemented by Trojan.Oddjob which attempts to stop any request from the user to log out of the online banking portal. This will keep the session alive and allow more time for transactions or other attacks. The trojan will intercept and block any GET/POST requests that contain predefined keywords in order to ignore the log-out request. This will keep the session open for the trojan. The config file contains strings like the following:

```
'*[NAME_OF_BANK].com*signout*'
'*[NAME_OF_BANK].com*logout*'
'*[NAME_OF_BANK].com/express/logoff.action*'
…
```

It is unclear why this was implemented in contrast to directly issuing a transaction, but it demonstrates that the attackers are trying new avenues and are coming up with creative ideas.

EVOLUTION ON THE DARK SIDE

Ten years ago many banking systems used static passwords as the only authentication method. They were easily broken by keyloggers and fake pop-up keyboards. Online banking systems have evolved, but so have banking trojans. Current banking trojan toolkits are well aware of the logic behind online banking sites and can check if there are multiple accounts joined together and if so choose the one with the highest balance. The trojans are capable of adjusting the account balance and removing any malicious transaction from the transaction history panel. This is done to maintain the illusion that nothing untoward has happened. With the use of web injects to dynamically modify the websites that are displayed to the user, a whole world of attacks became possible. Modifying the web traffic is still the most prominent and most successful attack. The whole process has become more dynamic; for example, money mules are loaded on the fly from the C&C server to ensure that they are always up to date and fresh.

On the code side, the evolution has not made big jumps. The code base has been adapted to work with more browsers and newer versions of the operating systems. Some variants will work fine even if the user has only limited privileges and is not using an administrator account. The binary itself now uses more obfuscation and usually encrypts the configuration file with algorithms like RC4. Carberp and others have started using rootkit features to hide themselves on the local system, sometimes going to the extent of creating a new hidden file system on disk to store the configuration files. The principle of injecting into the browser or hooking network APIs has mostly stayed the same. Most banking trojans support modules or allow plug-ins to be added. This allows for a simple extension of the feature set, and also allows single components that are detected by AV tools to be replaced. More threats, such as Tatanarg, have started to log if any security software is installed on the victim's machine and report it back to the C&C in order to fine-tune further attacks.

The attackers have definitely become more professional in offering and advertising their tools. Many banking trojans are now bundled with web attack toolkits, like Blackhole, for distribution. Often, ready-made installations can be rented as a service, requiring little to no IT knowledge from the buyer. Some authors have started to lock down toolkits with serial keys, because the trojan builder kits were frequently redistributed on underground forums for a cheaper price or even for free. The authors are obviously trying to protect their investment as well.

One of the weakest points for banking trojans is the botnet infrastructure. With only a few C&C servers, they risk being shut down in a coordinated takedown initiative. We have seen experiments with P2P networks to mitigate the risk on the attacker's side, but we believe that web scripts will remain the main control structure in the near future.

Some attackers have started using cloud services to host C&C servers or as drop boxes. Usually they use stolen credentials and piggyback on existing customers. Although, by definition this would mean that shutting down such a C&C in the cloud is not easy, most service providers are up to speed, respond quickly to takedown notices, and will suspend the fraudulent accounts if asked. As a result, the shift to the cloud does not make it more difficult to fight the malicious infrastructure and will not be a major change.

Two of the biggest challenges for the attackers are having enough suitable money mules and being able to transfer the money out of the accounts. These are issues which are on the process side and do not really fall into the coding aspect of banking trojans.

CONCLUSION

Malware attacks on online banking systems have been around for the last ten years and are still successful. There are around a dozen dedicated threats used in the wild, all performing similar attacks. The methods of the attackers have been slightly adapted but have not changed much over time. The biggest evolutions can be seen in the more robust command and control infrastructure, including P2P networks. We have also seen changes regarding how banking malware is offered, like renting preinstalled packages as malware-as-a-service or protecting it by serial keys. Custom web injects are offered on the black market. Furthermore, the obfuscation of the binaries themselves and the encryption of the configuration data has been improved over the years to make detection by anti-virus products more difficult and to hinder automated analysis.

From the standpoint of features used to bypass or break online banking security, very few new concepts have been introduced. We have seen experiments with *Firefox* extensions, but injecting to the browser or hooking network APIs are still the most common functions. Mainly because the currently deployed schemes do work on most occasions and the security features have not changed much either. Man-in-the-browser (MITB) attacks are still the most common and successful ones,

especially when combined with social engineering tricks. Pharming is rarely done and seems to be disappearing. The known concepts with web injects have been perfected, making the illusion complete with things like removing any malicious transaction from the history or changing the current balance to the previous level.

The shift towards more users using their mobile phone to perform online banking might bring a new growth of banking trojans for mobile platforms in the future, but the concepts of deception and MITB will most likely remain the same.

REFERENCES

[1] Wüest, C. Threats to Online Banking. http://www.symantec.com/avcenter/reference/threats.to.online.banking.pdf.

[2] ZeuS tracker. https://zeustracker.abuse.ch/statistic.php.

[3] http://pastehtml.com/view/1ego60e.html.

[4] Tarakanov, D. Big Brother. Securelist. http://www.securelist.com/en/blog/208193513/Big_Brother.

[5] http://pastehtml.com/view/b3cfzzw5l.html.

[6] Gutierrez, N. F. P. Trojan.Neloweg: Bank Robbing Bot in the Browser. http://www.symantec.com/content/en/us/enterprise/media/security_response/whitepapers/Trojan_Neloweg_Bank_Robbing_Bot_in_the_Browser.pdf.

[7] Wüest, C.; Florio, E. Firefox and Malware: When Browsers Attack. http://www.symantec.com/content/en/us/enterprise/media/security_response/whitepapers/firefox_and_malware.pdf.

[8] Doherty, S. Trojan.Tatanarg.B Careful! Symantec blog. http://www.symantec.com/connect/blogs/trojantatanargb-careful.

[9] Matrosov, A. Smartcard vulnerabilities in modern banking malware. Eset blog. http://blog.eset.com/2012/06/05/smartcard-vulnerabilities-in-modern-banking-malware.

[10] Direct Net Mobile Banking Questions & Answers. Credit Suisse. https://www.credit-suisse.com/ch/privatkunden/onlinebanking/doc/factsheet_direct_net_mobilebanking_en.pdf.

[11] Castillo, C. Android Malware Pairs Man-in-the-Middle With Remote-Controlled Banking Trojan. http://blogs.mcafee.com/mcafee-labs/android-malware-pairs-man-in-the-middle-with-remote-controlled-banking-trojan.