ISSN: 2454-9940

INTERNATIONAL JOURNAL OF APPLIED SCIENCE ENGINEERING AND MANAGEMENT

E-Mail :
editor.ijasem@gmail.com
editor@ijasem.org

www.ijasem.org

INTERNATIONAL JOURNAL OF APPLIED SCIENCE ENGINEERING AND MANAGEMENT

ISSN2454-9940

www.ijasem.org

Vol 18, Issue 1, 2024

A Sophisticated java based banking web application for smart and safe banking

P. VEERANNA, Department Of IT, SICET, Hyderabad

G. MAHIPAL REDDY, MOHAMMAD OSAMA RAYYAN, CHITTIMALLA PRAVEEN,KAMPETI SHIVA SAI, N VAMSHI RAM

UG Student, Department Of IT, SICET, Hyderabad

ABSTRACT

The Internet of today has become an integral part of our everyday life and the proportion of users expecting to be able to manage their bank accounts anywhere anytime is constantly growing. As such, Internet banking has come to age as a crucial component of any financial institution's multi-channel strategy.

Traditionally, information about financial institutions, their customers, and financial transactions areconsidered most sensitive. Doing such business via a public network consequently introduces new challenges for security and trustworthiness. Basically, any Internet banking system must solve the issues of authentication, confidentiality, integrity, and non-repudiation. This means it must ensure that only qualified people can access an Internet banking account, that the information viewed remains private and cannot be modified by third parties, and that any transactions made are traceable and verifiable. For confidentiality and integrity SSL/TLS (Secure Socket Layer) is the de-facto Internet banking standard while for authentication and non-repudiation no single scheme has become predominant yet.

Taxonomy of Internet Banking Authentication Methods

Internet banking systems must authenticate users before granting access to particular services. More precisely, the banking system must determine whether a user is, in fact, who he claims to be by asking the user to directly or indirectly prove knowledge of some sort of secret or credential. Based on the assumption that only an authentic user is able to do so, successful authentication eventually enables an authorized user to access his private information.

Expediently, all Internet banking authentication methods can be classified according to their resistance against two types of common attacks: offline credential stealing attacks (Figure 1) and online channel breaking attacks (Figure 2).

1. Offline credential stealing attacks aim at fraudulently gathering a user's credentials either by invading an insufficiently protected client PC by means of some malicious software such as a virus or trojan horse, or by tricking a user to voluntarily reveal his credentials through "phishing", that is, a combination of "spoofed" emails and mock-up web pages. Protection against malicious software can be achieved by a number of security precautions usually not strictly adhered to by the majority of private users: installing and maintaining a firewall and some up-to-date anti-virus software, regularly applying operating system and browser patches, and configuring the software appropriately. Phishing, in contrast, works by hijacking the trusted brands of well-known financial institutions and tricking users into entering their credentials into some faked web form. Common sense aside, phishers are able to convince up to 5% of customers addressed by some spoofed email to respond to them revealing their secrets[1]. This success rate is at least partially founded because most users actually do not know how to reliably identify a genuine banking server.
2. Online channel breaking attacks, such as the malicious "man-in-the-middle", the commercially motivated "market scorer" [13] or even the security-motivated "content inspector" [14], are even more sophisticated. Instead of trying to get hold of a user's credentials, messages betweenthe client PC and the banking server are unnoticeably intercepted, the intruder masquerading asthe server to the client and as the client to the server, respectively. Although the server is normally authenticated via a public -key certificate when a SSL/TLS session is established, oftentimes users are naively ignoring messages about invalid or untrusted certificates or, even worse, are fooled to trust online-generated fake server certificates from a nested intruder certification authority (CA). As a result the authenticated banking session could be hijacked or transaction data could silently be manipulated. In contrast to offline credential attacks that work decoupled from an actual user-initiated banking session, online channel breaking attacks do not necessarily compromise a user's credentials and in case require the user-initiated banking session to work on properly.

Based on the above taxonomy it now becomes possible to name the key properties that make Internetbanking authentication methods vulnerable against previously mentioned types of attacks (Figure 3).

Offline credential stealing attacks are effective only against schemes in which user credentials are valid for a rather long period of time (vulnerable against phishing), and stored or entered on a potentially insecure device such as the user's PC (vulnerable against malicious software). The most prominent example are static passwords (PW) which are assigned once and used repeatedly afterwards. Security hereby is simply based on the assumption that the password is non-trivial and kept secret, which in turn requires a trusted environment in which the password is used. Malicious software such as a virus or trojan horse, once installed on a client PC, can easily log all keyboard input and periodically email all data gathered to some predefined address. Phishing attacks are even easier to set up since only very limited context information (e.g., which bank a user is doing transactions with) is required. With a static password, an attacker then most likely is able to use the password fraudulently for some time without raising suspicion.

Secure Internet banking authentication solutions therefore at least rely on one-time passwords instead. The user is sent an ordered list of randomly chosen passwords (sometimes called a "scratch list") each of which is valid for one authentication only. Stealing such a one-time password while it is legitimately used makes no sense since it cannot be reused at all later on; all unused passwords must be kept secret, though. For convenience reasons, however, some users store their password list on their PC, effectively breaking the underlying security assumption and exposing their passwords to offline credential attacks. Malicious software then is able to steal the password list at any point in time, not only during authentication. Phishing is also possible yet, but may be slightly more difficult to make plausible if the banking server explicitly specifies which one-time password shall be used next. Fraudulently using one or more one-time passwords may eventually also be observed by the legitimate user.

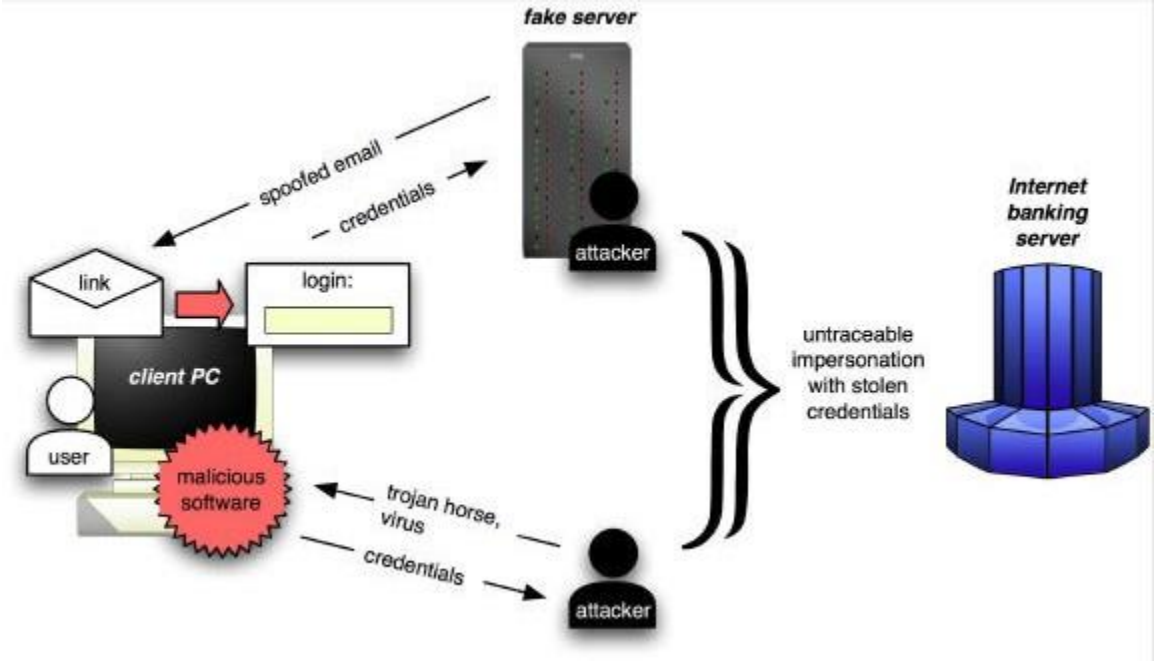

Figure 1: Offline Credential Stealing Attack Scenarios.

For crossing the offline-credential-stealing-attack boundary an authentication method must thwart both attacks by malicious software as well as phishing attacks. The former requires that credentials are never pre-exposed to some potentially insecure device such as the user's PC, while the latter is rendered infeasible by limiting the validity of a once exposed credential to a short period of time, effectively generating credentials on demand only. Both requirements are usually fulfilled by means of small microprocessor-based hardware tokens with a built-in display and some cryptographic key unique to the token. This key together with an additional source of entropy (e.g., the current time from a synchronized clock on the token, or a short-lived random challenge from the bank's server entered via a keypad on the token) then is used to generate short-time passwords that are valid for, say, 60 seconds only. Since these tokens are stand-alone devices neither directly nor indirectly exposed to the Internet, the user must manually copy the password from the display and enter it at the PC. Challenge/response tokens are

INTERNATIONAL JOURNAL OF APPLIED SCIENCE ENGINEERING AND MANAGEMENT

ISSN2454-9940

www.ijasem.org

Vol 18, Issue 1, 2024

commonly considered slightly more secure than timer-based tokens since the additional source of entropy is short-lived, non-deterministic and ideally also bound to an account number.

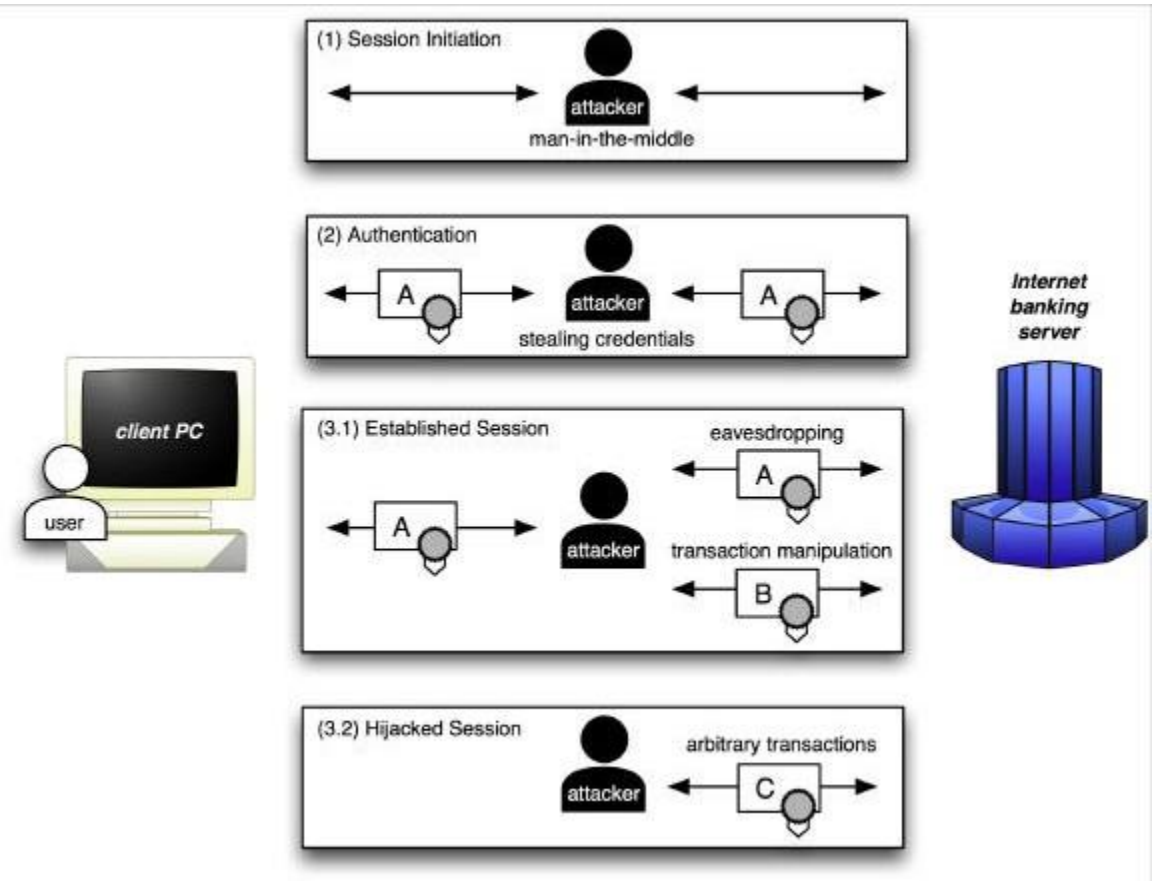

Figure 2: Online Channel Breaking Attack Scenarios.

Authentication based on a hardware token Public-Key Infrastructure (PKI) also avoids the risk of offline credential stealing attacks against insufficiently secured PCs. Most notably, as of today, these schemes are also the only ones that effectively cross the online-channel-breaking-attack boundary independently of the user behaviour. PKI makes use of asymmetric cryptographic algorithms such as RSA (Rivest Shamir Adlemann) or ECC (Eliptic Curve Cryptography). Initially each user is fit with a pair of matching private and public keys for which some trusted authority issues a matching digital certificate. The certificate attests that the user name is actually associated with the given public key, and that the user is holding the corresponding private key. The private key and certificate are used then to establish a mutually authenticated SSL/TLS channel between, for instance, the user's PC and the bank's server, effectively eliminating online channel breaking attacks. The critical issue here is the protection of the users private key against malicious software. If stored in a so-called "soft token", a password-encrypted file on the user's PC, the password and consequently also the private key would be exposed to offline credential attacks. The private key therefore must be stored on some tamper-resistant hardware token such as a microprocessor-based smart card, potentially exposing only private-key related functionality. Today's smart cards implement a variety of hardware and software countermeasures thwarting physical as well as logical attacks against the card itself. Consequently, stealing a private key from a smart card becomes almost impossible. A potential point of attack might result from using the smart card, though. In most cases the card is protected by some PIN which must be sent to the card to unlock it; only if this happens the private-key functionality become available. Yet entering and sending the PIN to the smart card via the PC exposes the PIN and consequently also the private-key functionality to malicious software. Despite being very sophisticated and systematic, such attacks are perfectly feasible and can be eliminated only by introducing a certified tamper-resistant smart-card reader equipped with keypad and display. This way sensitive operations requiring user interaction such as entering the PIN are moved from the potentially

INTERNATIONAL JOURNAL OF APPLIED SCIENCE ENGINEERING AND MANAGEMENT

ISSN2454-9940

www.ijasem.org

Vol 18, Issue 1, 2024

exposed PC to the trusted reader device which interacts with the user and the smart card only directly through its own secure interfaces.

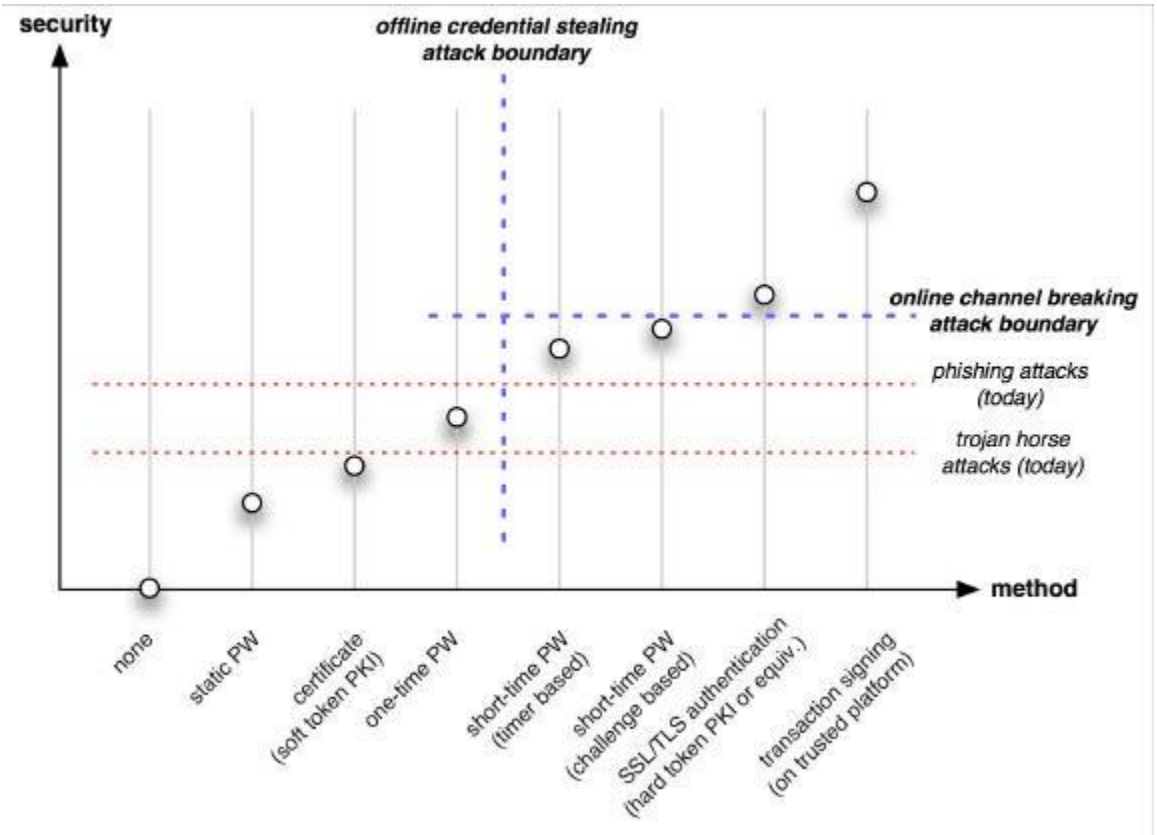

Figure **3:** Taxonomy of Internet Banking Authentication Methods.

In the following sections we present two state-of-the-art Internet banking authentication schemes based on challenge/response: the first one using short-time passwords from an offline hardware token, and the second one making use of a hardware-token based PKI and a FINREAD secure smart-card reader.

Short-Time Password Solution

Considering today's pervasiveness of malicious software (viruses, trojan horses) and phishing attacks, any Internet banking solution must be resistant against offline credential stealing attacks. For this we propose a challenge/response-based short-time password authentication method using symmetric cryptography in combination with a hardware security module (smart card) and an offline (stand alone) smart-card reader (Figure 4). This solution provides convenient mobility for "road warriors" who want to do Internet banking at any time from anywhere, not just from their homes. At the core of this scheme is a smart card that is personalized with a randomly chosen symmetric cryptographic key, for example, a 3DES or A ES (Data/Advanced Encryption Standard) key, and a strictly monotonic counter; the smart card is protected by a PIN. The use of a symmetric scheme here is crucial for enabling the output of the algorithm to be shortened to an appropriately user-convenient and yet sufficiently guessing-attack resistant size. The user communicates with the card via an offline smart-card reader equipped with a small display and keypad. The keypad on the reader is used to enter both the PIN to temporarily open the smart card for further processing, and afterwards an n-digit challenge. For communication with the banking server, standard web

INTERNATIONAL JOURNAL OF APPLIED SCIENCE ENGINEERING AND MANAGEMENT

ISSN2454-9940

www.ijasem.org

Vol 18, Issue 1, 2024

browsers (Microsoft Internet Explorer, Netscape Navigator) are employed to provide the user interface; all web pages comprise standard HTML code only.

User authentication then works as follows:

1. The user connects to his Internet banking server via SSL/TLS with server-side authentication; this way the user may ensure to be connected with a genuine banking server by explicitly validating the server certificate.
2. The user claims his identity by entering his account number on the bank's login form and, in turn, the banking server displays an n-digit challenge, asking for a matching m-digit response.
3. The user opens his smart card by entering the corresponding PIN on his smart-card reader before entering the given challenge. The smart card then calculates the matching response by encrypting the challenge and the incremented on-card login counter with its symmetric cryptographic key and encoding the result as an appropriately presentable response string.
4. The user manually copies the shown response to the bank's login form to be checked by the bank's authentication server redoing the same calculation independently. Since the login counters on the smart card and on the server may diverge (e.g., if a user playfully calculates some responses), the server tries to synchronize its local counter within a small range of, say, 32 counter values.

This scheme successfully thwarts offline credential stealing attacks. Since the user's credentials are stored on a tamper-resistant smart card and are only accessed through an offline smart-card reader, there is no way for malicious software to get hold of the user's symmetric cryptographic key or related functionality. Phishing attacks also don't work because there is no way for an attacker to know which challenge will be given next by a genuine banking server and because challenges are short-lived and bound to an account number. Without additional measures, however, the scheme is not suitable for crossing the online-channel-breaking-attack boundary independent of the user behavior. Given the high level of expertise required to launch such online channel breaking attacks, the remaining risk may be acceptable though in view of the benefits gained, especially regarding mobility.

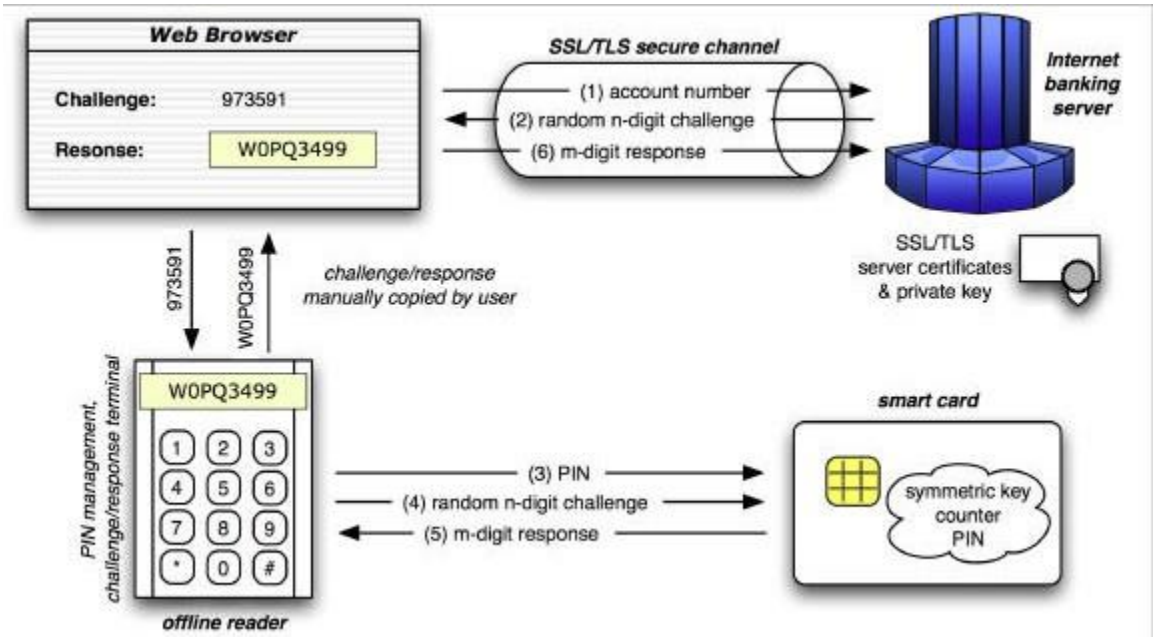

Figure 4: Overview Short-Time Password Solution.

Technically the most elegant way for crossing the online-channel-breaking-attack boundary with such a scheme would be an extension to SSL/TLS supporting symmetric in addition to asymmetric SSL/TLS client authentication. Instead of today's default SSL/TLS procedure, where a protocol-data-dependent challenge can only be signed with a user's private key, a challenge/response round trip with explicit user

interaction would be required, involving the use of a short-time password authentication scheme as just described. Most notably, such a change would be rather non-intrusive. Only the SSL/TLS certificate request would have to be extended to allow the server to ask for an appropriately derived part of the SSL/TLS challenge to be displayed on the user screen, and an appropriate response or short-time password to be gathered from the user keyboard. The challenge and response length and format hereby are four independent parameters (e.g., a six-digit numeric character challenge and an eight-digit alphanumeric character response). Recall that, by signing a protocol-data-dependent challenge the authentication response becomes channel-specific, i.e., only valid for the server to which the SSL/TLS channel has been set up, and therefore perfectly resists online channel breaking attacks.

Certificate-Based Solution

With a more stationary setup, crossing the online-channel-breaking-attack boundary independent of the user behavior becomes mandatory [15]. For this we present a two-stage, PKI-based Internet banking authentication solution that is characterized by consistently applying open standards and making use of a programmable certified secure smart-card reader connected to a potentially exposed PC (Figure 5).

Similar to the challenge/response short-time password solution the user receives a smart card acting as the secure token for his Internet banking account. In this case the smart card includes an advanced microprocessor chip which supports RSA public-key cryptography. Furthermore, it holds a smart-card operating system compliant with the JavaCard(tm) specification publicly available from Sun Microsystems [2]. JavaCard has become the de facto standard in smart-card operating systems in recent years. It represents a platform-independent multi-application run-time environment based on a Java virtual machine. Since JavaCards can be sourced from many different manufacturers, JavaCard applications (so-called applets) are not only independent from the hardware platform itself but also from the card manufacturer.

For use in a PKI environment, an applet must be loaded onto the smart card which stores key pairs plus matching certificates and generates digital signatures. The PKCS#15 Cryptographic Token Information Format Standard [3], defined and maintained by RSA Laboratories, specifies how information such as keys, certificates, or PINs have to be structured on a smart card. This way, with a JavaCard applet implementing PKCS#15, another level of independence between the on-card application and the PC software communicating with the smart card is achieved. The user's smart card or, more precisely, the PKCS#15 application on the card, then is personalized with an RSA key pair along with a matching certificate issued by a CA operated by the bank. The PKCS#15 application protects the private-key functionality with a PIN in a way such that signatures can only be generated if the valid PIN has been presented beforehand. Furthermore, the user must be in possession of a FINREAD smart-card reader connected to his PC. Similar to JavaCard in the world of smart cards, FINREAD is a set of open technical specifications defining the properties of a secure smart-card reader device (FCR) [4]. An FCR must have certain physical properties such as tamper resistance, a secure display, and a secure keypad. Tamper resistance applies in particular to the reader credentials since an FCR holds its own set of cryptographic keys and even may perform RSA calculations. Furthermore, the firmware of an FCR provides a multi-application run-time environment based on a Java virtual machine. Each FCR then can host platform-independent Java applications, so-called FINREAD card reader applications (FCRA), which can only be loaded in a strictly controlled way. Most of the keys hosted by the reader are required for reader management operations such as the loading of FCRAs, the maintenance of the reader firmware and the exc hange of the keys themselves. One RSA key, however, is available to be used by FCRAs for application-level purposes. The main idea of FINREAD is to enforce that the smart card is solely used on a secure trusted reader device to reinforce the security level provided by the smart card and thus to achieve strong end-to-end security. For our certificate-based solution, we use a dedicated FCRA along with an appropriate FINREAD card reader identification application (FCRIA) which are loaded onto the user's FCR to secure the authentication process. More specifically, the FCRIA does not allow transparent smart- card access, that is, it is not possible to communicate directly with the smart card from a PC application. The FCRIA strictly controls all requests targeting the card, and all security critical operations such as PIN

INTERNATIONAL JOURNAL OF APPLIED SCIENCE ENGINEERING AND MANAGEMENT

ISSN2454-9940

www.ijasem.org

Vol 18, Issue 1, 2024

verification or signature generation are blocked. Those actions are processed by the FCRA only and require explicit user approval on the FCR. In addition to a standard programming interface for FCRAs/FCRIAs, FINREAD also defines an open standard client API for interfacing PC application to FCR/FCRAs in a manufacturer independent way [4, 6].

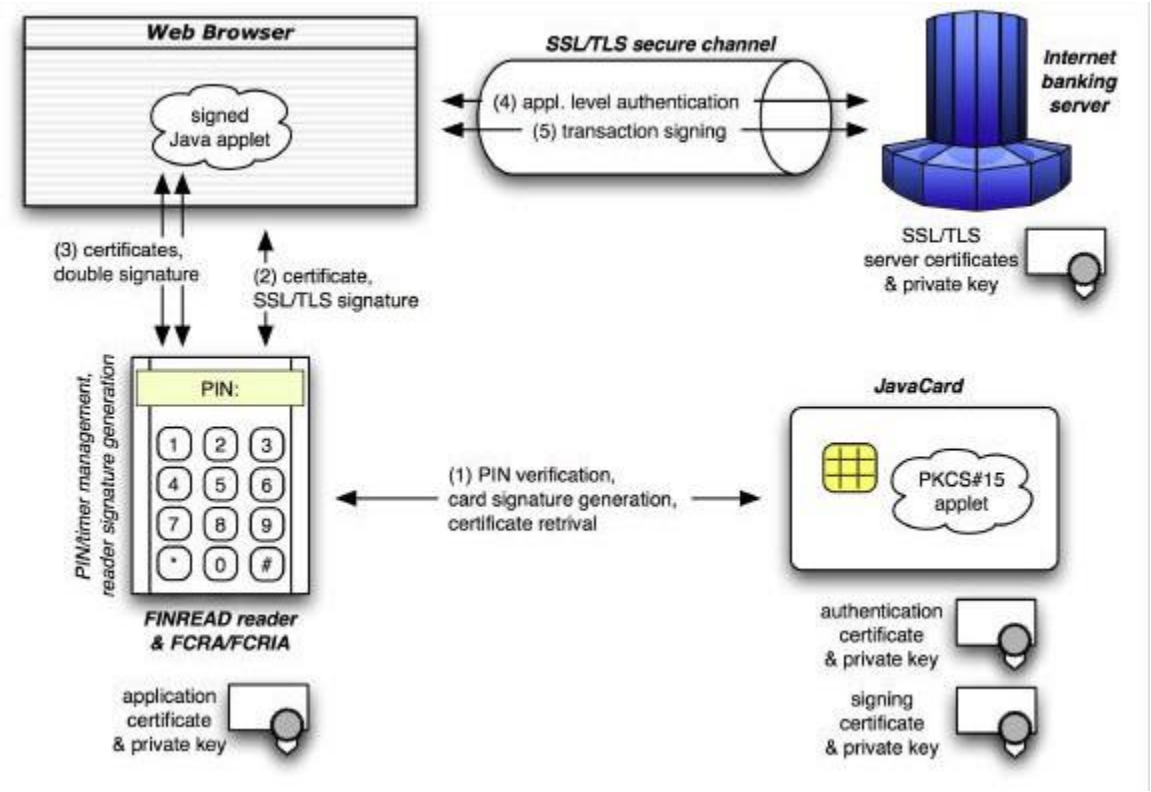

Figure 5: Overview Certificate-Based Solution.

As the user interface for the Internet banking system, again standard web browsers (Microsoft Internet Explorer, Netscape Navigator) are employed. The web pages comprise standard HTML and JavaScript code. Furthermore, communicating with the card reader from within the web browser requires a mechanism to access native code such as a pre-installed DLL. Here we embed a signed Java applet into the web pages which provides this mechanism via the Java Native Interface (JNI) keeping our solution browser independent (yet not platform-independent). For making the user's certificates visible for the browsers SSL/TLS implementation, though, two more software components must be installed on the client PC. Netscape Navigator provides a secure token interface in accordance with the PKCS#11 open standard from RSA Laboratories [5]; such a PKCS#11 library therefore can be used to connect Netscape with the FCR/FCRA. Microsoft Internet Explorer, in contrast, uses the Microsoft Crypto API to access keys and certificates which requires a cryptographic service provider (CSP) to be installed as interface between Microsoft Internet Explorer and the PKCS#11 library.

User authentication then works as follows:

1. A mutually authenticated SSL/TLS channel between the user PC and the bank's web server is established. For this, first a SSL/TLS session without client authentication is set up. A Java applet running in the web browser then checks if a FINREAD reader is available and a valid smart card is present in the reader's smart card slot. Otherwise, the user is requested to insert a valid smart card. Once the card is available, its certificates become visible in the web browser and the server initiates an SSL/TLS renegotiation (via an SSL/TLS Client Hello Request), this time with client authentication. During SSL/TLS client authentication the generation of a digital signature response on the protocol-data-dependent challenge is required at the client side. The FCRA on the card reader detects this and requests the user to input his PIN. Given

that the PIN is valid, the FCRA initiates a signature generation with the authentication key on the card to complete the SSL/TLS client authentication. An encrypted and mutually authenticated SSL/TLS session has now been established over which all the following communication traffic will be sent. The SSL/TLS key exchange together with the protocol-data-dependent client authentication exclude online channel breaking attacks, as further explained at the end of the previous section.

2. An additional user authentication then is performed at the application layer. A random challenge is sent to the client which again is forwarded to the FCRA for signature generation. The FCRA reuses the card's authentication key to sign the challenge and then double-signs the signature from the card with the readers application key. Both signatures are submitted together with the corresponding card and reader certificates to the server for verification. If both signatures are valid and the same client key has been used both for the SSL/TLS and application-layer authentication, the banking server is sure that a genuine card has been used in a genuine reader, eventually excluding also offline credential stealing attacks. To prevent the card from remaining unlocked once the PIN has been presented and successfully verified, the FCRA maintains timers and counters appropriately limiting the availability of the keys on the card.

The scheme effectively thwarts both offline credential stealing as well as online channel breaking attacks. Because of the FINREAD reader that intercepts all calls to the smart card, malicious software cannot silently access the smart card and get hold or make use of the user's credentials. Moreover, the protocol-data-dependent client authentication eliminates both phishing and online channel breaking attacks. Most notably, the crucial SSL/TLS server certificate verification is implicitly ensured here, at least to some extend, by the channel-specific (server certificate dependent) digital signature response.

Transaction Signing Option

Once an authenticated channel has been established between the user and the bank, the authorized user may be able to freely work with his account carrying out all kind of transactions such as transferring funds or trading shares. Most sophisticated attacks, however, specifically manipulating transaction data on the client PC, are theoretically still conceivable. To thwart such content manipulation attacks both of the above challenge-based solutions, in contrast to timer-based solutions, provide the option to sign alphanumeric transaction data before submitting it to the server.

In the certificate-based solution the transaction data is sent to the FCRA and the critical details appear on the secure display of the FINREAD reader. Given that the user explicitly approves the transaction via the secure keypad, the FCRA double-signs the transaction with the cards signature key (a second RSA private key that has been personalized into the PKCS#15 applet, cf. Figure 5) and the readers application key, similar to the application-level authentication step where a random challenge was signed. By executing critical operations on the trusted reader device and by involving the user via the trusted reader interfaces, content manipulation attacks can be eliminated. Furthermore, this method allows for tracing and verifying individual transactions and thus also provides means for non-repudiation.

In the short-time password solution a similar proctection is achievable, although less convenient, by having the user generate a dedicated password (message authentication code) in response to a user-intelligible transaction-related challenge, e.g., a beneficiary account number that the user may check. This represents an effective protection against content manipulation attacks. Since the underlying symmetric key is shared with the bank, however, it does not represent an appropriate means for non-repudiation.

Related Work

The authentication schemes and attacks introduced in the first chapter represent the standard of knowledge discussed in various publications dealing with user authentication [9]. However, most of them solely

provide an overview of schemes and corresponding attacks not attempting to draw a security landscape by relating them to each other in a sensible way. This is what the taxonomy presented in Figure 3 provides in the context of the particular application domain Internet banking.

Short-time password solutions based on a password generating hardware token are available from various manufacturers such as RSA Security, ActivCard or VeriSign. The RSA's SecureID solution [10] is the most prominent example. It consists of a small device including a LCD display and one button the user can press to initiate the calculation of the next short-time password. In contrast to the solution presented in this article it is timer-based and does not use a smart card to ensure state-of-the-art tamper resistance and easy device personalization. Furthermore, as it is generally not equipped with an alphanumeric keypad the short-time password generation functionality can neither be PIN protected nor can it be extended for transaction signing. Challenge-response based solutions seem to be available, too, but are rarely used on a large scale. Obviously convenience comes before security at this point.

Only a few banks have decided to make use of public key cryptography for their Internet banking system, mostly to avoid setting up and maintaining a public key infrastructure (PKI). One example where a PKI solution is in production is the Migrosbank [11]. Here a smart card is used to securely store a RSA private key and sign some data in the context of a challenge/response authentication protocol. In contrast to the FINREAD-based solution presented here these type of PKI solutions mostly use simple card reader devices, not equipped with a secure keypad, display or cryptographic capabilities. Also, the authentication protocol is often preformed on application level via a server authenticated SSL/TLS channel, rather than using the smart card to perform SSL/TLS client authentication. Advanced solutions utilizing FINREAD are beginning to emerge only now. Within the European-Union-funded project Trusted FINREAD [4] a remote banking pilot was carried out to demonstrate basic functionality and interoperability of the FINREAD platform. However, with the usage of SSL/TLS client authentication and double signatures allowing to authenticate the card as well as the reader device the solution presented in this article is certainly superior and can be considered one of the strongest internet banking authentication schemes today. Apparently, most of today's solutions using keypad-equipped readers, for instance, solutions compliant with the Home banking computing interface (HBCI) [12] are using the reader mainly to implement secure PIN entry.

Conclusions

Internet banking has come to age as an arms race between financial institutions and public network attackers. Yet with the latest authentication schemes presented in this paper, banks can potentially take a clear advantage. Both solutions presented in this article offer high security against common attacks. The certificate-based scheme is currently the only one additionally thwarting online channel breaking attacks, but we hope that this article will motivate leveraging this important security property to non-PKI based mobile authentication solutions as well. Interestingly, more sophisticated client-platform attacks conceivable in the future such as, for instance, content manipulation attacks can simply be thwarted by activating the ready-to-use transaction signing option. From a user perspective, the main difference between the two solutions is that the first one supports mobility whereas the second one is more convenient. With the use of JavaCard, however, it is easily possible to integrate both on one smart card, this way providing the full range of options, usage and security-wise.

In contrast to commercial token solutions, where keys are often generated abroad by the token manufacturer, smart-card-based solutions mo st noticeably allow for a financial-institution-controlled key management within the national jurisdiction; symmetric keys being generated in-house and directly transferred via an end-to-end secured channel into the smart card and asymmetric keys being directly generated on the smart card. Well established national smart card personalization infrastructures (typically used for credit or debit card personalization) further guarantee the scalability of the solution and thereby also its perfect suitability for large Internet e-banking customer populations.

We conlude by noting that the above analysis underlies the current UBS Switzerland e-banking authentication strategy and that both proposed solutions have been successfully implemented In this context.

The short-time password solution is in use at UBS since 2002, with currently more then 500'000 users. Despite its inevitable impact on user convenience, the solution has always been appreciated and perceived by the majority of the UBS e-banking customers as a significant security enhancement, compared to the scratch-list based one-time password solution they were using before. By now, as customers have become more familiar with the solution, a market research and analysis recently testified an unexpected high level of general customer appreciation covering security but, more importantly, also overall efficiency and convenience.

The certificate-based solution has been piloted at UBS in 2004, with about 100 internal users only. Yet missing readiness of the reader technology and the client PC software platform let to the decision to postpone spreading this solution to customers. With regard to changing legislation and the eventually expected natural spreading of e-ID's among customers, the solution none the less rema ins a highly attractive and valuable alternative for the future [7-8].

References

[1] Anti-Phishing Working Group. http://www.antiphishing.org

[2] JavaCard Platform Specification. Sun Microsystems. http://java.sun.com/products.javacard

[3] PKCS#15: Cryptographic Token Information Format Standard. RSA Laboratories.

[4] FINREAD Specification. FINREAD Consortium. (accepted as work item 1.17.34 for ISO/IEC JTCS1/SC17). http://www.finread.com

[5] PKCS#11: Cryptographic Token Interface Standard. RSA Laboratories.

[6] PC/SC Specification Version 2.0. http://www.pcscworkgroup.com/specifications/specdownload.php

[7] Secure user authentication over a communication network. UBS Patent, Europe: EP-87 458; US: 10/237,080

[8] Secure user and data authentication over a communication network. UBS Patent, Europe: EP -87 903; US: 10/235,936

[9] Richard E. Smith. Authentication: From Passwords to Public Keys. Addison Wesley, 2001.

[10] RSA SecureID Token. RSA Laboratories.

[11] M-Card smart. Migrosbank Switzerland. http://www.migrosbank.ch/de/Private/KartenZahlungsverkehr/MCardMCardSmart.htm.

[12] FinTS – Financial Transaction Services. http://www.hbci-zka.de/english/index.html

[13] http://www.pcworld.com/news/article/0,aid,118757,00.asp

[14] http://www.microdasys.com/

[15] Bruce Schneier. Two-Factor Authentication: Too Little, Too Late. Communications of the ACM vol 48, no.4, April 2005.