

Risk Assessment of a National Security Infrastructure

In Norway, BankID is the banking industry's public-key infrastructure (PKI) of choice for authenticating Internet customers. But do BankID's differences from standard PKIs make it a riskier choice? This assessment, based on both publicly available information and usage experiences, addresses that question.



KJELL J. HOLE,
ANDRÉ N.
KLINGSHEIM,
LARS-HELGE
NETLAND,
YNGVE ESPELID,
THOMAS
TJØSTHEIM,
AND VEBJØRN
MOEN
*University of
Bergen*

A public-key infrastructure is a collection of hardware, software, processes, and people that together provide security services based on public-key cryptography. Many countries, including Norway, Sweden, Denmark, Finland, Estonia, Austria, Belgium, and Canada, have introduced large-scale security frameworks implementing PKIs. In particular, nationwide PKIs can provide strong authentication services, reducing the risk of security breaches compared to frameworks that use, for example, fixed or one-time passwords. However, it's both difficult and costly to design and implement a large PKI correctly, so there's a real danger of ending up with a flawed solution.

The Norwegian banking industry's PKI, BankID, primarily authenticates Internet banking customers. Currently, the country's banking industry is pushing for BankID to become a national ID infrastructure that government agencies and commercial companies would use to authenticate individuals and provide legally binding digital signatures with a high degree of non-repudiation.

In previous articles, we analyzed Norwegian Internet banking and automatic teller machine systems.^{1,2} Here, we take on BankID and examine how it differs from a typical PKI based on the X.509 ITU-T standard. We then offer a qualitative risk assessment³ of its user authentication and discuss the non-repudiation service. Because the Norwegian banking community declined to share technical information about BankID, we could evaluate it only from the outside and were unable to assess important aspects of the system, such as contingency planning

and disaster recovery. Our evaluation, which we completed in January 2008, is based on publicly available descriptions of the BankID architecture and design, as well as our personal use of the system.

Overview: Public-Key Infrastructure

A sender and receiver who plan to use symmetric-key cryptography to communicate securely over a public channel must first establish a common secret using some other means of communication. Establishing such secrets quickly becomes impractical when many parties want to communicate securely over a public network such as the Internet. The discovery of public-key cryptography in 1976 showed how senders and receivers could set up secure communication without first having to use a separate channel to establish a secret. Today, PKIs use public-key cryptography to facilitate secure communication over the Internet. While there are several types of PKIs, most commercial PKIs are based on the X.509 ITU-T standard and typically share several essential features.

Keys, Signatures, and Certificates

An X.509 PKI uses a pair of asymmetric cryptographic keys—one private and one public. Only the owner has access to the private key, whereas the public key is available to all users.⁴ The private key is stored in an encrypted file on the user's computer—or, better yet, on a tamper-resistant smart card. For example, to encrypt a message to Alice, anyone can use Alice's

public key, but only Alice can access the private key to decrypt the message.⁵

Digital signatures offer an alternative to handwritten signatures. For example, Bob might use his private key to digitally sign an important message or document before sending it to Alice. During the cryptographic signing procedure, a value—denoted as the digital signature—is calculated over the message. Alice verifies Bob’s digital signature by applying a cryptographic procedure that takes as input Bob’s public key, the received message, and the signature. If somebody tampered with the message after it was signed, then the verification will fail; otherwise, we assume Bob signed the message because he alone had access to the private key.⁵ An X.509 certificate binds a public key to an identifier—such as a personal name, an assigned user number, or a Web address—that points to the user or Web site with the corresponding private key.

Commercial PKIs, such as BankID, use different key pairs for digital signatures and encryption/decryption. For simplicity, we don’t differentiate between these pairs here.

PKI Architecture

A new PKI user requests a certificate from a registration authority, providing the information required. The RA verifies the information and sends a certification request containing the user’s public key to the certification authority. The CA then generates the certificate and signs it with its own private key. A Web site owner initiates a similar procedure to obtain a Web site certificate.

A certificate is valid only for a given time period set by the CA. The CA revokes a certificate before it expires if it becomes apparent that the certificate holder’s public key should no longer be used. The CA also keeps track of all revoked certificates. Certificate revocation frequently occurs when a company terminates a customer relation. Also—and more importantly—the CA revokes a certificate immediately if the corresponding private key is compromised. Often, CAs maintain a digitally signed certificate revocation list (CRL) containing unique references to the revoked certificates and the dates and reasons for the revocation.

A single PKI might have multiple CAs.⁴ For simplicity, we consider a strict, two-level CA hierarchy. First, a single root CA on the zeroth level has a special self-signed certificate containing the CA’s own public key and a signature generated with the CA’s own private key. The root CA generates and signs certificates to the level-one CAs, which issue certificates to the PKI’s users.

Transitive Trust Model

It’s useful to view the concept of trust as a measure of reliance between two entities. In the setting of

an X.509 PKI, trust is a binary concept consisting of “trust” and “no trust.” An entity X trusts another entity Y when X assumes it knows exactly how Y behaves. Users trust a CA, for example, when they assume that the name and public-key binding in an issued certificate is correct. If X trusts Y and Y trusts Z, then X also trusts Z. So, if users trust a root CA, they also trust a level-one CA. If they don’t trust the root CA, they won’t use the PKI services.

In a PKI, the starting point of all trust is the root CA. In a two-level CA hierarchy, certificate path processing extends trust from the root CA to a level-one CA. Consider a case in which two different level-one CAs issue certificates to Bob and Alice. To verify Alice’s certificate, Bob first builds a path of certificates back to the root CA. The path consists of Alice’s certificate, the certificate of the level-one CA that issued it, and the root CA certificate. Using the root CA’s public key, Bob then verifies the level-one CA certificate’s signature. Next, he extracts that CA certificate’s public key and uses it to verify the signature of Alice’s certificate. Because Bob trusts the root CA, he now assumes that the content of Alice’s certificate is correct.

Authentication

Authentication establishes an understood degree of confidence that an identifier actually refers to a user or a Web site.⁶ If the confidence is high, then authentication is strong.

PKI authentication is based on certificates, the corresponding private keys, and a source of revocation information, such as a CRL. When Bob wants to authenticate Alice, he first asks for her certificate and then uses the CRL to verify that the certificate hasn’t been revoked. He also confirms Alice’s ownership of the public key by building and verifying a certificate path to a trusted root CA. Bob then asks Alice to sign a random number, or *challenge*, with her private key. If Bob verifies the signed challenge using Alice’s public key, then he assumes he’s communicating with Alice. To authenticate Bob, Alice would carry out the same protocol.

Authentication strength relies on three factors: a well-tested authentication protocol, such as the Transport Layer Security (TLS) protocol⁷; secure private-key storage; and the computational difficulty of calculating the private key. To maintain authentication strength over time, CAs must update their CRLs often and make the lists continuously available to both parties during the authentication process.

Non-Repudiation: The Legal View

Non-repudiation protects a person against another person falsely claiming that a communication didn’t take place.⁸ The two most commonly discussed types of non-repudiation are

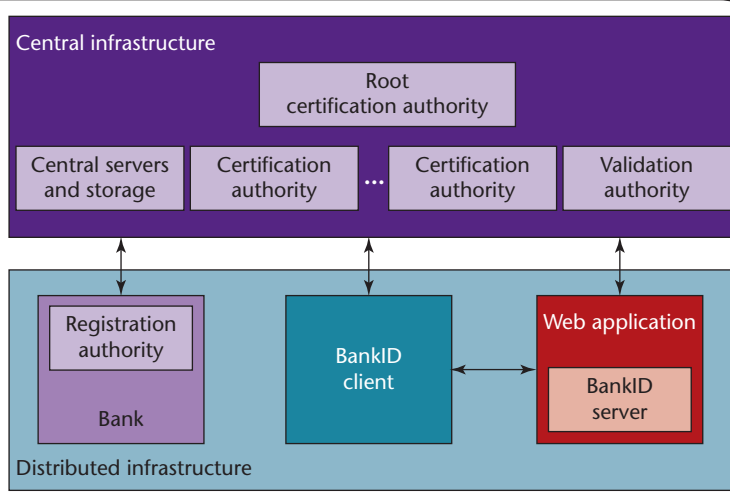


Figure 1. The BankID architecture's information flow. The architecture consists of two main parts: a central infrastructure and a distributed infrastructure.

- *non-repudiation of origin*, which aims to reveal people who falsely deny having sent a message; and
- *non-repudiation of delivery*, which aims to reveal people who falsely deny having received a message.

There are several views on how to understand non-repudiation. From a legal viewpoint pertinent to the Norwegian banking community and its customers, non-repudiation consists of the ability to convince a third party that a specific message originated with or was delivered to a certain person. Credible evidence to that end is needed to persuade a judge, jury, or arbitrator. The evidence's quality and presentation determine the degree of non-repudiation.

Organizations can build a non-repudiation service that uses digital signatures on top of a standard PKI. They can obtain a high degree of non-repudiation by combining basic PKI services with the correct combination of legal and technical non-repudiation protocols. It's also essential that at least one trusted third party collects, validates, time stamps, signs, and stores relevant information. The third party must be able to withstand pressure from the communicating parties and present the non-repudiation evidence in an unbiased manner during a conflict.

A high level of non-repudiation implies only that it will be possible to show, with high probability, that a person actually digitally signed a particular document even if he or she later denies it. Hence, non-repudiation doesn't remove a person's legal right to refute a signature. Also, the burden of proof is on the party wanting to rely on a digital signature.

BankID Explained

BankID is a PKI developed and owned by the Norwe-

gian banking community. It's different from most PKIs because users' key pairs are stored and used on a central infrastructure, not on smart cards or user computers.

Certificate Types

BankID offers three different types of user certificates:

- a personal certificate for regular users,
- an employee certificate for users representing a company or an organization, and
- a certificate for Web sites.

All three certificates are based on X.509 version 3.⁴ In addition to these certificates, CAs, RAs, and other entities in the BankID infrastructure have their own certificates.

Architectural Overview

As Figure 1 shows, the BankID architecture^{9,10} is divided into two main parts. The Norwegian Banks' Payment and Clearing Centre (also known as BBS) operates the central infrastructure, which contains CAs, a certificate validation authority (VA), central storage facilities for cryptographic keys and certificates, and functionality for digital document signing. The second part, the distributed infrastructure, comprises the BankID server, which is part of a Web application; the RA at a user's bank; and the BankID client running on the user's computer.

Central Infrastructure

The central infrastructure includes a two-level hierarchy with one root CA (owned by the Norwegian Financial Services and Saving Banks Associations) and multiple level-one CAs. Although the level-one CAs are part of the central infrastructure, each is owned by a separate bank or group of banks. A bank uses its CA to issue certificates to its own customers. The root-CA certificate is valid for 26 years, while the level-one CA certificates are valid for 12 years.¹⁰

The VA utilizes CRLs from the level-one CAs to determine if certificates have been revoked; this validation service is available to both the BankID server and the client in Figure 1.

BankID Server

An entity using BankID—such as an online store or Internet banking site—incorporates a BankID server into its Web application. The server software is available in both the C and Java programming languages. The BankID server stores its certificate and private key in a hardware security module (HSM) or an encrypted Public-Key Cryptography Standard #12 file.¹¹ HSMs also provide Web applications with dedicated hardware for processor-intensive cryptographic operations.

Bank RA

Each bank operates its own RA software, which is typically integrated into customer service software. New BankID customers can request a certificate from the RA either online or in person at their local branch office. The RA then sends the certification request through a TLS tunnel to the central BankID infrastructure, which starts initializing a new customer record. First, the central infrastructure generates at least one public-private key pair. It then stores the private key in a central database and sends the public key and the RA request to the CA belonging to the customer's bank. The CA then generates a new certificate, which is stored on the BankID central infrastructure.

BankID Client

The BankID client is a Java applet embedded in all BankID affiliates' login Web pages. The applet runs in customers' Internet browsers each time they use the BankID system. Hence, no software or information is permanently stored on a customer's computer. The applet drives both the BankID authentication and digital signature protocols.

BankID Authentication Procedure

BankID's user authentication protocol is complicated. Here, we present only enough information to make the risk assessment comprehensible; a more detailed protocol description is available elsewhere.¹²

As Figure 2 shows, BankID divides the authentication between a user and a Web application into two parts. The first part is a two-factor authentication procedure. In it, BankID authenticates users to the central infrastructure and lets them control access to their own centrally stored cryptographic keys. The second part is a challenge-response protocol between the BankID client and server. All communications between the entities in Figure 2 are executed through TLS tunnels.

Authentication and key access control. Because users can have multiple BankID certificates issued by different banks' CAs, they first enter their Norwegian social security number (SSN) into the BankID client. The central infrastructure responds with a list of all their BankID affiliations, letting users choose the one they want. The client then prompts users for a one-time personal identification number, which the central infrastructure verifies. A hardware token, or PIN calculator, generates this one-time PIN. (Users sometimes need a fixed PIN to activate the PIN calculator.) Finally, the client prompts users to enter a fixed password that the central infrastructure uses to gain access to the their private key.

These procedures are very similar to a traditional two-factor authentication mechanism: they're based on something you have (the PIN calculator) and

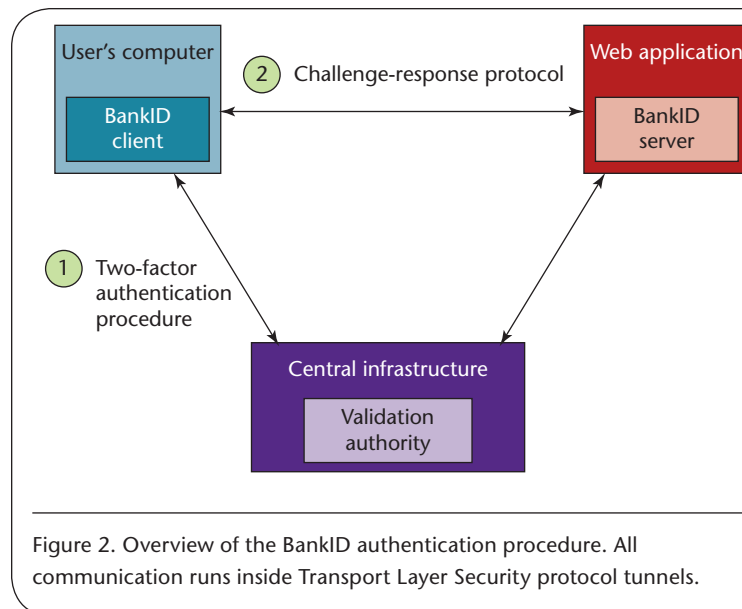


Figure 2. Overview of the BankID authentication procedure. All communication runs inside Transport Layer Security protocol tunnels.

something you know (a fixed password and, perhaps, a fixed PIN that activates the calculator).⁶

Challenge-response protocol. Once the central infrastructure accesses the user's cryptographic keys, it can complete the authentication by executing the challenge-response protocol. During this process, the central infrastructure carries out the cryptographic functions for the BankID customer. It uses the VA to verify the certificates for the BankID server and the customer.

BankID differs from a typical X.509 PKI in that it centrally stores asymmetric keys and centrally executes cryptographic operations. Given this, the BankID client must transmit a one-time PIN and a fixed password over the Internet to the central infrastructure.

Non-Repudiation

BankID aims to provide non-repudiation of both origin and delivery. Unfortunately, it keeps secret nearly all information about its legal and technical non-repudiation protocols and how it stores non-repudiation evidence. We do know, however, that signed documents contain the signed data, the signatures of the parties, and the VA request results at the time of signing.⁹ Here again, BankID differs from a typical X.509 PKI in that it doesn't use a trusted third party to establish non-repudiation information for dispute resolution.^{9,10}

Authentication Service

We define a risk as the possibility of suffering financial loss due to an attack on a system's service.³ We now assess the risks associated with two simple attacks on BankID's authentication service as used by Norwegian Internet banks.

We divide each risk assessment into two steps. The

first step is based on a technical analysis of an attack. We use a Microsoft model, Dread (for damage potential, reproducibility, exploitability, affected users, discoverability)¹³ to assign low, medium, or high severity levels to these five important risk aspects, which cover the impact from and the difficulty of exploiting relevant system vulnerabilities. Dread assumes the existence of threats that are willing and able to exploit vulnerabilities. Given this, the second step determines an attack's likelihood using publicly available threat information. Because this information is quite limited, we present the results of the two steps separately.

DDoS Attack Based on Account Lockout

This attack exploits the fact that the BankID client accesses the central infrastructure's cryptographic functionality after the user has provided the correct SSN, a one-time PIN, and a fixed password. In our experiments, if customers enter a correct SSN and a wrong PIN three times, BankID shuts down central infrastructure access and customers must contact their bank to reopen their account. (Access is also denied if customers type in a correct SSN and PIN but enter an incorrect password multiple times.)

It follows that it's possible to automate an account lockout attack. We ran a simple proof of concept using an executable script that starts Internet Explorer 7, downloads the BankID client, and simulates a user logging in with the correct SSN and then entering the wrong PIN three times.

Step 1: Dread analysis. The account lockout attack is specific to systems using traditional two-factor authentication over the Internet. Currently, BankID uses two-factor authentication—inherited from the previous generation of Norwegian Internet banks—despite the fact that we have documented the existence of efficient distributed denial-of-service (DDoS) attacks and communicated them to the Norwegian banking industry.¹ Because the attack is well known, discoverability is high in our analysis.

Norwegian SSNs have a well-defined structure, so it's possible to generate a large set of SSNs containing SSNs belonging to BankID customers (as well as SSNs that don't belong to customers).¹ A small program can then run through the SSNs set and close down customer accounts, as described earlier. Further, intruders can spread the attack over many computers and thereby construct a DDoS attack at the application layer. Such an attack is difficult to stop if it uses numerous computers (> 10,000).^{1,14} For DDoS attacks at lower network layers, intruders must transmit a large amount of dummy traffic over a long period of time to block account access. At the application layer, however, they can simply attempt to log in to each account a few times. Because this efficient DDoS attack on the

authentication procedure has the potential to deny all users access to BankID, the authentication procedure represents a single point of failure.

As long as the BankID authentication procedure remains unchanged, it will remain relatively easy to repeat an application-layer DDoS attack. Given that a simple program can automate account closings, that all BankID users might be affected, and that intruders can easily repeat the attack, our analysis indicates high exploitability, a high number of affected users, and high reproducibility.

More than 1 million of the 2.3 million Internet banking customers in Norway have already moved to BankID. If, as the Norwegian banking industry envisions, most of the remaining customers also move to BankID, it might well have more than 2 million users. Because they'll all rely on the same infrastructure, a successful DDoS attack would have severe economic consequences. Roughly 2 million people would be unable to access their accounts, transfer funds, or pay bills, and some online stores that use BankID would be unable to sell goods. Furthermore, if BankID also becomes a national ID used by government agencies, the consequences of a DDoS attack would be even more severe. Hence, the damage potential is high.

Our analysis clearly shows that there's a high risk associated with an application-layer DDoS attack on the BankID system.

Step 2: Threat analysis. It's been argued that the likelihood of a DDoS attack against a Norwegian online banking system is small because no one group is sufficiently motivated to carry out such an attack. Traditional hackers might be reluctant to carry out a DDoS attack on BankID because they don't see a financial upside and might fear the inevitable investigation by Norwegian national security agencies. However, other groups—such as those that strongly oppose Western society in general and Norway in particular—might find BankID a tempting target to create chaos in the Norwegian financial system. While we've yet to see public reports of a DDoS attack on BankID, a future political conflict might induce such an attack. The massive and much discussed DDoS attack on Estonia is an example of a politically motivated attack that inflicted significant losses on a nation's economy.

Phishing/Man-in-the-Middle Attack

In March 2007, we created a proof-of-concept phishing/man-in-the-middle (MITM) attack against BankID using well-known attack techniques. In late 2007 and early 2008, changes to BankID mitigated the risk associated with this type of attack.

In our proof-of-concept attack, we altered two URL parameters to make the BankID client communicate with the BankID server and central infrastruc-

ture through a hacker-controlled proxy server. The URL parameters let us turn off the TLS tunnel between the BankID client and the proxy. Consequently, we didn't have to provide the proxy with a certificate. To initiate the attack, we used phishing email¹⁵ to trick BankID customers into downloading modified HTML code together with the unaltered BankID client. During a proof-of-concept demonstration to the Financial Supervisory Authority of Norway, we accessed accounts in two randomly chosen Internet banks that used BankID. (A detailed description and analysis of these exploits is available elsewhere.^{12,16}) Our analysis indicated that malicious software in users' browsers could also initiate this attack.

Step 1: Dread analysis. It took approximately 100 hours to implement our attack. Because the exploit requires a skilled programmer, we assess it at medium exploitability.¹³ The attack involves running the MITM proxy and crafting and distributing phishing emails to potential victims, making it a highly reproducible attack. Also, we pointed out the weakness to the public in 2007, resulting in high discoverability.

Relying on phishing limits the exploit's number of victims. Most users won't fall for the phishing scheme, mandating a low ranking for affected users. (If BankID gets 2 million users and 1 percent of them fall for a phishing email, then the number of victims is 20,000.) However, a successful attack could lead to significant losses for the affected customers and some losses for the banks in terms of financial liabilities and impaired reputation. The risk level for damage potential is therefore medium.

Overall, we associated a medium risk for the combined phishing/MITM attack on online BankID banks between May 2007 and January 2008.

Step 2: Threat analysis. According to Norway's Financial Supervisory Authority, professional criminals attacked six Norwegian online banks—not based on BankID—during the second half of 2006 and the start of 2007. Apparently, the attackers used tailor-made Trojan horse software to take control over Internet banking sessions after customers had completed a two-factor authentication process. Funds from the compromised accounts were transferred to other online accounts whose owners, called “money mules,” then transferred the money abroad. It's therefore reasonable to conclude that criminals are motivated to exercise phishing/MITM attacks similar to the one we describe.

The Non-Repudiation Service

Our discussion of BankID's degree of non-repudiation focuses on the strength of its user authentication, the lack of a trusted third party, and the BankID community's refusal to provide information about the unusual

cryptographic keys storage on the central infrastructure. We also discuss the potential risk to a user when a Web site owner, such as a merchant or an online bank, falsely claims that the user digitally signed a document.

Given our inability to access information about certain parts of the system, as well as uncertainty regarding the legal implications of the unknown system details, we were unable to perform a complete risk assessment. Instead, we highlight questionable aspects of BankID.

User Authentication Vulnerabilities

If a PKI's user authentication is too weak, a skilled hacker can digitally sign a document using the victim's identity. It can be difficult for victimized users to prove that they didn't sign a document. When PKIs offer non-repudiation services, this only increases the problem for these unfortunate users because the merchant has access to collected non-repudiation evidence—including the digital signature—showing that the user signed the document, when in fact it was a hacker misusing the user's identity. Hence, strong user authentication is needed to achieve a high degree of non-repudiation.

It's well known that the strengths of different authentication techniques vary.⁶ Password-based authentication is rather weak, for example, while two-factor authentication is stronger. A traditional PKI—utilizing a request-response protocol based on the users' locally stored asymmetric cryptographic keys—can provide even stronger authentication, partly because it's unnecessary to transmit traditional secrets (such as passwords and PINs) over the Internet.

As Figure 2 shows, BankID's user authentication is a hybrid solution, consisting of a two-factor authentication procedure followed by a request-response protocol. How strong is this hybrid authentication technique? To answer, we start with the fact that hackers can always download the BankID client (Java applet). If hackers can break or circumvent the two-factor authentication such that the BankID clients get access to the central infrastructure's cryptographic functionality (and thus, indirectly, to a user's asymmetric keys), then the applet can complete the request-response protocol. Hence, if hackers can compromise the two-factor authentica-

BankID doesn't use a trusted third party to achieve non-repudiation, despite the fact that most non-repudiation protocols require it.

tion, they can compromise the user authentication. In other words, the request-response protocol doesn't add to the user authentication's strength, and BankID's hybrid user authentication is no stronger than traditional

two-factor authentication. As we described earlier, BankID's two-factor authentication was vulnerable to practical attacks in 2007.

No Trusted Third Party

BankID doesn't use a trusted third party to achieve non-repudiation, despite the fact that most non-repudiation protocols require it.⁸ BankID-member banks have strong financial relationships with both users and merchants that own Web sites. This is especially true when those Web sites are Internet banking sites; the banks own both those sites and the complete BankID infrastructure, giving them tremendous control over the financial operations requiring non-repudiation.

The BankID community apparently concentrated on creating a non-repudiation service to protect Web site owners from dishonest customers; it put much less effort into protecting customers from dishonest Web site owners, including the banks themselves.

Security-through-Secrecy Policy

BankID stores a user's private key on the central infrastructure, which is controlled by the Norwegian Banks' Payment and Clearing Centre. This key is used only inside an HSM.⁹ Because private keys must be available only to users to achieve a high degree of non-repudiation,⁴ the central key storage raises several questions:

- How are the private keys generated on the customers' behalf and stored in the HSM without anyone possibly learning the keys' value?
- How can customers provide a fixed password to activate the authentication and signing functionality without anyone else obtaining the password?
- Is a customer's network connection made directly to the HSM to avoid MITM attacks from rogue insiders?
- Can a rogue insider access the keys by exploiting the HSM's API, modifying existing code, installing new malicious code, and/or modifying server configurations?

The Norwegian banking community has refused to share any information about the central storage of BankID keys. This security-through-secrecy policy is unfortunate; the lack of information makes it difficult for users (or their lawyers) to challenge claims about the non-repudiation service during a dispute.

Weak and Unfair Non-Repudiation

The user authentication's limited strength restricts BankID's degree of non-repudiation. Its non-repudiation is further limited by the lack of a trusted third party and the security-through-secrecy policy, both of which give Web site owners an advantage

over the customer during a conflict involving signature repudiation. BankID's non-repudiation service is therefore both weak, because its degree of non-repudiation is unsatisfactory, and unfair, because it favors one party over another during a conflict. Users signing documents, possibly concerning assets of significant financial value, are thus subject to an unknown risk.

In a dispute, performing a thorough risk assessment requires a team with expertise in both computer science and Norwegian law. A user's financial costs during a judicial conflict might be significant, but the likelihood of a conflict is small because, in practice, the use of digital signatures in BankID is still limited. Finally, non-repudiation protocols described in the literature⁸ show that it's possible to significantly improve BankID's weak and unfair non-repudiation service.

In late April 2008, BankID remained vulnerable to DDoS attacks at the application layer, and its user authentication strength needed further investigation. However, according to a letter from the Norwegian Financial Services and Saving Banks Associations to the Norwegian Parliament commenting on our research (which was verified by an independent researcher¹⁷), customer risk is zero because the banks will cover financial losses due to attacks. Nevertheless, the non-repudiation service's lack of a trusted third party and the banks' security-through-secrecy policy put the customers at a disadvantage during potential conflicts involving digital signature repudiation. This problem could grow if BankID becomes a national ID infrastructure widely used by both government agencies and commercial companies. □

References

1. K.J. Hole, V. Moen, and T. Tjøstheim, "Case Study: Online Banking Security," *IEEE Security and Privacy*, vol. 4, no. 2, 2006, pp. 14–20.
2. K.J. Hole et al., "Lessons from the Norwegian ATM System," *IEEE Security and Privacy*, vol. 5, no. 6, 2007, pp. 25–31.
3. A. Calder and S.G. Watkins, *Information Security Risk Management for ISO27001/ISO17799*, IT Governance Publishing, 2007.
4. C. Adams and S. Lloyd, *Understanding PKI*, 2nd ed., Addison-Wesley, 2003.
5. W. Stallings, *Cryptography and Network Security*, 4th ed., Prentice Hall, 2006.
6. S.T. Kent and L.I. Millett, eds., *Who Goes There? Nat'l Academies Press*, 2003.
7. S.A. Thomas, *SSL and TLS Essentials*, Wiley, 2000.
8. J. Zhou, *Non-Repudiation in Electronic Commerce*, Artech House, 2001.

9. The Norwegian Banks' Payment and Clearing Centre (BBS), *BankID FOI White Paper*, release 2.0.0, 2006 (in Norwegian).
10. Bankenes Standardiseringskontor, *Norsk BankID Sertifikatpolicy for Banklagrede Kvalifiserte Sertifikater Til Personkunder*, version 1.1, 2005 (in Norwegian).
11. RSA Laboratories, *PKCS #12 v1.0: Personal Information Exchange Syntax Standard*, 1999.
12. Y. Espelid et al., "Robbing Banks with Their Own Software—An Exploit against Norwegian Online Banks," *Proc. 23rd Int'l Information Security Conf. (SEC 2008)*, Springer, 2008, pp. 63–77.
13. J.D. Meier et al., *Improving Web Application Security: Threats and Countermeasures*, Microsoft, 2003.
14. J. Mirkovic et al., *Internet Denial of Service*, Prentice Hall, 2005.
15. L. James, *Phishing Exposed*, Syngress, 2005.
16. Y. Espelid et al., "A Proof-of-Concept Attack against Norwegian Internet Banking Systems," *Proc. 12th Int'l Conf. on Financial Cryptography and Data Security (FC 08)*, LNCS 5143, Springer Verlag, 2008, pp. 197–201.
17. K. Gjosteen, "Weaknesses in BankID, A PKI-substitute Deployed by Norwegian Banks," *Public Key Infrastructure*, LNCS 5057, Springer Verlag, 2008, pp. 196–206.

Kjell J. Hole is a professor in the Department of Informatics, University of Bergen, Norway. His research interests include

risk management of computer systems. Hole has a PhD in computer science from the University of Bergen. He is a member of the IEEE and the IEEE Computer Society. Contact him at kjell.hole@ii.uib.no.

André N. Klingsheim is a senior security analyst at NoWires Group AS. He has a PhD in computer science from the University of Bergen. He is a member of the IEEE and the IEEE Computer Society. Contact him at ank@nowiresgroup.com.

Lars-Helge Netland is a senior security analyst at NoWires Group AS. He has a PhD in computer science from the University of Bergen. Contact him at lh@nowiresgroup.com.

Yngve Espelid is a software developer at Bouvet ASA. He has a PhD in computer science from the University of Bergen. Contact him at yngve.espelid@bouvet.no.

Thomas Tjøstheim is a product manager and security analyst at EDB Business Partner, Norway. He has a PhD in computer science from the University of Bergen. Contact him at [thomas.tjostheim@edb.com](mailto:tjostheim@edb.com).

Vebjørn Moen is a governance and security manager at GE Money Bank, Norway. He has a PhD in computer science from the University of Bergen. Contact him at [vebjoern.moen@ge.com](mailto:moen@ge.com).

computing | now

ACCESS | DISCOVER | ENGAGE

Let us bring technology news to you.



<http://computingnow.computer.org>
Subscribe to our daily newsfeed