



Research

Nordea mobile bank v2.3.2 for Android: Multiple vulnerabilities

31st of May 2012

Juan J. Güelfo

CEO, IT-security consultant and
IT-security researcher at Encripto AS

Table of contents

About Encripto AS.....	3
Timeline and revision history.....	3
Disclaimer.....	3
License.....	3
1. Introduction: Why Nordea Bank.....	5
2. Findings.....	5
2.1 App vulnerable to Man In The Middle attacks.....	5
2.2 Hostile payment creation and approval using the victim's bank account	6
2.3 Sessions are not destroyed properly when the user logs off.....	7
2.4 Use of deprecated encryption cipher for data storage, with an insecure encryption mode.....	7
2.5 Information leaks due to insecure data storage	7
3. References	10

About Encripto AS

Encripto AS is a Norwegian company which specializes in information security. We mainly work with penetration testing and education within information security. Encripto works with big and small companies, both from public and private sectors. Quality is priority number one for everything we do, and details matter to us.

Encripto AS is committed to information security. We do research to discover trends, new vulnerabilities and better ways to mitigate them. We believe in acting as good internet citizens to the industry, whether you are a provider or a user.

You can read more about us at <http://www.encripto.no>

Timeline and revision history

- **31st of May 2012**
The report is published (Report public disclosure).
- **8th of May 2012**
The vendor releases Nordea Mobilbank v2.3.3, with patches which address the most important security issues.
- **4th of May 2012**
Nordea Bank contacts Encripto AS for more information about the vulnerabilities. Coordinated disclosure is agreed to the end of May.
- **2nd of May 2012**
Initial version (Report non-public disclosure). Encripto AS contacts Nordea Bank and sends a copy of the vulnerability report. Nordea confirms to have received the report.
- **30th of April 2012**
Vulnerabilities are discovered by the researcher.

Disclaimer

The material presented in this document is for educational purposes only. Encripto AS cannot be responsible for any loss or damage carried out by any technique presented in this material. The reader is the only one responsible for applying this knowledge, which is at his / her own risk.

No servers, external clients or individuals were breached during this research. The vulnerabilities were found using the researcher's bank account.

Any of the trademarks, service marks, collective marks, design rights, personality rights or similar rights that are mentioned, used or cited in this document is property of their respective owners.

License

This document is licensed under the terms of the Creative Commons Attribution ShareAlike 3.0 license. More information about this license can be found at <http://creativecommons.org/licenses/by-sa/3.0/>

1. Introduction: Why Nordea Bank

Smartphones and tablets are very popular these days, especially in Norway. It is usual to find companies which develop their own apps for these devices, and banks are no exception.

Nordea Bank advertises its banking application widely, so their customers can benefit of an easy and portable banking solution. Since the researcher owns an account at this bank, he decided to check whether the mobile bank application was keeping a decent security and privacy levels.

The researcher will not perform any kind of evaluation in this document. This project was done on his own initiative, and the findings will be presented in an objective way, so the vendor will be able to address the problems.

2. Findings

Nordea's mobile bank application for Android can be downloaded from Android Market for free. When installed, this application will provide different banking services.

The application is divided at least in two parts:

1. The actual mobile application which is installed in an Android smartphone or tablet.
2. A web application which performs banking operations, such as fund transfers, payments and view payment list.

When the user decides to access the mobile bank, the actual mobile application (presented previously in point 1) will act as a browser, which connects to <https://nettbank.bbs.no/mobnordea> (presented in point 2). This URL is the actual web application that hosts the banking web application.

The following vulnerabilities have been found on Nordea's mobile bank v2.3.2 for Android. This version was the latest one available on Android Market at the time the research was conducted.

At the same time, one of the vulnerabilities presented on this document affects the banking web application directly.

2.1 App vulnerable to Man In The Middle attacks

The mobile application is vulnerable to Man In The Middle (MITM) attacks. The application never validates the server SSL/TLS certificate before establishing a secure connection with the bank.

During the MITM attack, the attacker stands between the banking server and the client mobile application. At this point, the attacker can provide a self-signed SSL/TLS certificate which is under the attacker's control. In this way, the attacker will be able to tamper the SSL/TLS encrypted connection.



Fig. 1: Structure of a Man In The Middle attack

Since the client mobile bank application does not check the validity of the SSL/TLS certificate before establishing a secure connection, the application goes ahead and negotiates an encrypted connection with the attacker.

The connection with the attacker is established automatically. No error messages or warnings are displayed to the user. At this point, the attacker is able to get access to the information transmitted from the mobile application to the banking server (and vice versa) in plain text (the attacker controls the encryption).

Since the attacker can get access to the information, he will have access to the authenticated cookie / session ID which belongs to the client. In this way, the attacker can hijack the user's banking session. In other words, the attacker will be able to access and control the user's bank profile.

MITM attacks can potentially happen in any kind of network, but they have become especially popular in public networks, such as wireless hotspots at airports, coffee shops or train stations.

2.2 Hostile payment creation and approval using the victim's bank account

Provided the attacker performs a MITM attack successfully, and the user session is hijacked, the attacker can create a payment to any account of his choice. In other words, the attacker can steal funds from the victim's account.

The payment creation routine is divided in two steps:

1. Payment creation, which requires no validation.
2. Payment verification, which requires a code from the user's security card / security token.

Since the attacker controls the victim's session, the first step can be performed without any problems. However, the transaction must be validated by the user in step 2.

This validation can happen as soon as the victim submits a secure code coming from the victim's security card / security token. When performing the MITM attack, the attacker is in the middle of the connection. This gives him access to all the information that the mobile application is sending to the banking server, before it actually arrives to the server. Therefore, the attacker can extract and use the security tokens when the user submits them to the banking server in a regular transaction.

This attack would happen following these steps:

- **Step 1:** The attacker performs a MITM attack and hijacks the user session.
- **Step 2:** The attacker creates a payment to any account of his choice.
- **Step 3:** The user submits a verification code from his security card. This may happen due to the user is about to approve a payment or transaction created by him / her.
- **Step 4:** The attacker intercepts the security code, and forwards an empty value to the server. Since the attacker manipulated the value submitted by the user with an empty value, the user gets a regular verification error from the banking server.
- **Step 5:** At this point, the security code intercepted by the attacker can be used for approving the attacker's transaction.
- **Step 6:** The attacker's payment is approved and included in the victim's payment list.

2.3 Sessions are not destroyed properly when the user logs off

The bank web application at the server side (<https://nettbank.bbs.no/mobnordea>) does not manage session information properly. Sessions are not destroyed when the user pushes the log off button.

This allows an attacker to remain logged on the user's bank profile even if the user signs out.

2.4 Use of deprecated encryption cipher for data storage, with an insecure encryption mode

The mobile application stores different kinds of data in the smartphone's memory. The information which is encrypted by the mobile application, such as Social Security Numbers, uses the deprecated DES cipher in Electronic Codebook (ECB) mode.

After the information has been encrypted, it is base64-encoded and finally stored in the smartphone's memory.

Example:

The Social Security Number 11223344556 was DES encrypted in ECB mode and Base64 encoded by the mobile application. The result is stored in a file called MAN.xml:

`/AV//FbZ2c2T6YC36BEeBA==`

The encryption key is stored as a string in plain text, which is located at the smartphone's memory, stored in a file named strings.xml together with other data. The key looks like this:

```
<string
name="secret_key">CyetVerdUpDoquajrecJewyeodMilmiasFicsOkerorjomWorthoakmuodd
OrKevnacfevaljyadruflaifKigejyutkotDotUtneenFigCockvabovopsilvathyat6oabowgav
jefnecucyuWofquenDybCihabdityibfikwiAnMechFakinnabvicIs7WyoxoddyeckDaktijOadr
odChagvahaykCacShulsEypDoykHofhanyerOrc~SluRoanlahysuhirzAwshEkCorbikcacteitF
ucyithoogudryumHenjaxnokVobCul10kopFalbAkTageb6gra1HepyifDytsEpveudFiQuabjano
acjapseawsifMuwat</string>
```

By performing a Base64-decoding and DES-decrypting (ECB mode) the results with the given key, it was possible to retrieve the original Social Security Number.

An online DES decryption tool can be found at tools4noobs.com^[1]

More information about the insecure Electronic Codebook (ECB) encryption mode, and a comparison with other secure modes, can be found at Encripto's blog^[2].

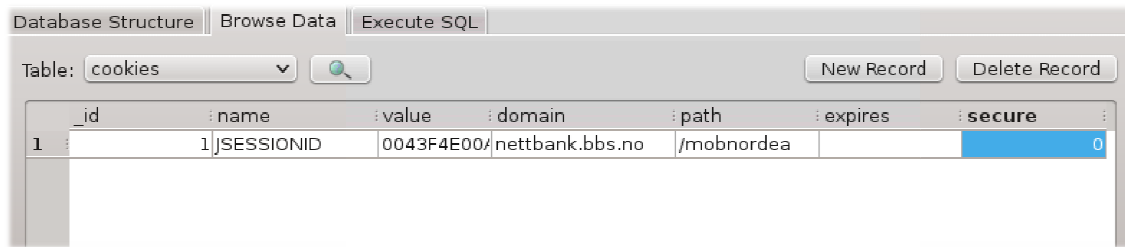
2.5 Information leaks due to insecure data storage

The mobile application handles information, which is stored in the smartphone's memory in plain text.

This information contains a full description of payments created by the user, cookie information with session IDs, and history of URLs visited by the user while he / she is connected to the banking server. Also, information about the user's account statement is stored in plain text. Any person or malware (installed in the user's smartphone) could potentially have access to this information.

Specifically, the database webview.db contained the following important information:

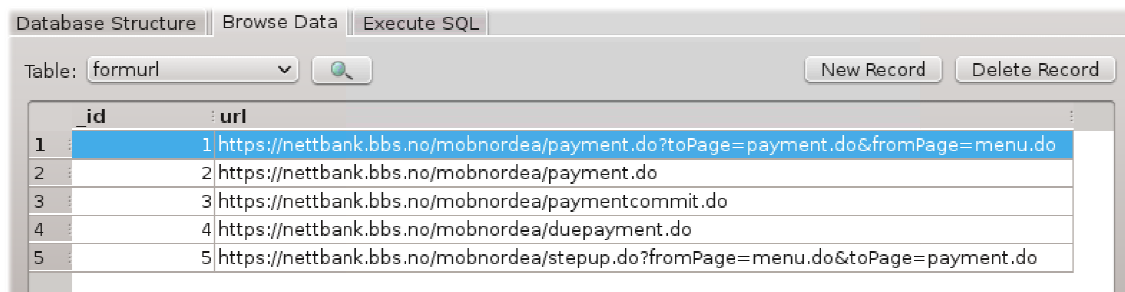
- **Table: cookies (user session)**



_id	name	value	domain	path	expires	secure
1	SESSIONID	0043F4E00	nettbank.bbs.no	/mobnordea		0

Fig. 2: The mobile application keeps session information in a table, which is stored unencrypted in the smartphone's memory.

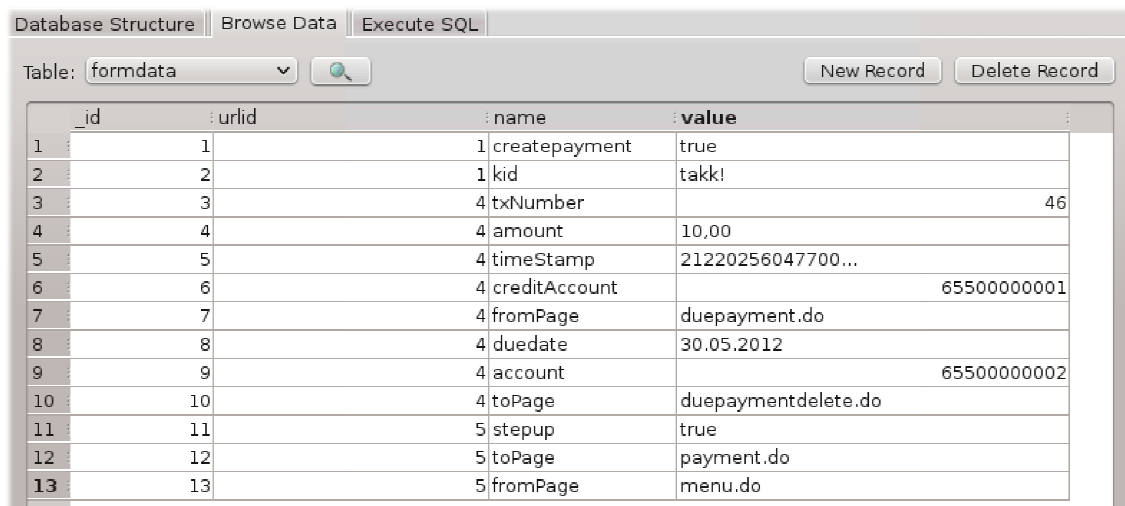
- **Table: formurl (browsing history)**



_id	url
1	https://nettbank.bbs.no/mobnordea/payment.do?toPage=payment.do&fromPage=menu.do
2	https://nettbank.bbs.no/mobnordea/payment.do
3	https://nettbank.bbs.no/mobnordea/paymentcommit.do
4	https://nettbank.bbs.no/mobnordea/dupayment.do
5	https://nettbank.bbs.no/mobnordea/stepup.do?fromPage=menu.do&toPage=payment.do

Fig. 3: The pages visited with the mobile application are stored unencrypted

- **Table: formdata (transaction data)**



_id	urlid	name	value
1	1	createpayment	true
2	2	kid	takk!
3	3	txNumber	46
4	4	amount	10,00
5	5	timeStamp	21220256047700...
6	6	creditAccount	65500000001
7	7	fromPage	dupayment.do
8	8	duedate	30.05.2012
9	9	account	65500000002
10	10	toPage	dupaymentdelete.do
11	11	stepup	true
12	12	toPage	payment.do
13	13	fromPage	menu.do

Fig. 4: Transaction data, such as payments, are stored unencrypted.

Due to privacy reasons, the account numbers and the timestamp in this screenshot have been filled out with dummy data.

In addition to these tables, there were two other (httpauth, password) which could contain important information. However, no more research was done at this point.

Moreover, the mobile application stores data from the user's bank profile, such as account balance and account statement. This information can be found as part as the cache information that the mobile application keeps in the smartphone's memory (webviewCache folder).

Example:

The actual information retrieved from the cache has been filled out with dummy data due to privacy reasons.

```
<li class="txElement" id="3">
<h3>
  <span class="date">01.04.2012</span>
  <span class="amount">-100,00</span>
</h3><h4>
  <em class="type">KTO.REG</em>
  <span class="text"></span>
  <span class="text"></span>
  <span class="text">OVERFØRT NETTBETALING 65500000001</span>
</h4></li>
```

3. References

[1] Online decryption tool from Tools4noobs.com

http://www.tools4noobs.com/online_tools/decrypt/

[2] Encrypto256: When encryption is not enough. Juan J. Güelfo.

<http://encrypto256.encrypto.no/2012/03/when-encryption-is-not-enough.html>