# Online Banking Security

Fabian Alenius  
Uwe Bauknecht  
May 17, 2009

## Contents

1. Introduction  

2. Secure Communication  
   2.1 Password authentication  
   2.2 One-time Passwords  
   2.3 SSL  
   2.4 Security Tokens  

3. Implementations  
   3.1 Chip Authentication Program (CAP)  
   3.2 RSA SecurID  
   3.3 Open Authentication (OATH)  

4. Conclusion  

References
1 Introduction

Today, online banking services are becoming more and more common. The acceptance among the general populace has risen significantly since the introduction. The term online banking was actually coined originally in the 1980’s, it referred to banking done remotely by using a terminal connected to a phone line [8]. Today however the term is mostly associated with banking services offered over the Internet.

To get some idea of how fast online banking is growing, the UK has seen an 174% increase in the numbers of user between 2001 and 2007 [1]. This hasn’t been completely unproblematic however, as fraud has gone up considerably as well [2]. This has prompted banks to introduce stronger security measures. In this paper we will start with a basic authentication scheme and continually improve it as we identify potential attacks.

2 Secure Communication

Most Internet shopping sites use usernames and passwords to authenticate its users, so called ‘password authentication’. They are typically more concerned with the validity of the credit card than the identity of the user. This will be our starting point.

2.1 Password authentication

In our fictitious example we have a user Alice who wishes to login to her bank. We also have a vicious attacker Eve who is trying to steal Alice’s hard-earned money. The bank is using a username and password to protect Alice’s account but no encryption. This scheme is obviously vulnerable to a snooping attack as illustrated in Figure 1. One way to improve security is by employing One-time Passwords.

![Diagram of Eve snooping Alice’s password and subsequently logs in.](image.png)

Figure 1: Eve snoops Alice’s password and subsequently logs in.
2.2 One-time Passwords

One-time passwords (OTPs) are, like the name suggests, passwords that are used only once. By constantly altering the password, it becomes substantially harder to mount an attack. Eve can no longer login using Alice’s username and password since the server will not accept the same password twice.

Another common use of OTPs is as so-called transaction authentication numbers (TAN). With the TANs, instead of spending an OTP whenever the user logs into the bank’s website, a normal username/password combination is used. The user will have to input an OTP/TAN only, if he wants to perform an actual transaction like transferring money from one account to another.

The OTPs have to be transfered to the user in some way, the easiest of which is to send them by mail. An alternative more secure way is to force the user to pick them up at his local bank office after providing proper identification. A sample code scratch card can be seen in Figure 2.

![Figure 2: A code scratch card with OTPs](image)

The OTPs can be implemented using a hash-chain[3]. The bank starts with an initial secret $S_0$ and constructs $n$ OTPs using a hash function $H()$ to compute $S_{n+1} = H(S_n)$. The bank then stores $S_n$ and sends the remaining $n - 1$ codes to the customer. In Table 1 we can see how the OTPs are used.

All is not good however, OTPs are still vulnerable to man-in-middle-attacks, although it has become more difficult. If Eve can manipulate the traffic between Alice and her bank in such a way, that she can not only listen in to the connection, but actively block and forge messages, then she can simply take Alice’s OTP, prevent her from contacting the bank possibly displaying a fake error message and login herself. See Figure 3. What we need to do is encrypt the traffic so that Eve can no longer read the password.
2.3 SSL

SSL is an abbreviation of Secure Socket Layer and is a protocol designed to provide security and data integrity. The steps in setting up a SSL connection are depicted in Figure 4 and described in 2. SSL supports a wide range of algorithms, some very strong and some weak. For example Handelsbanken, a Swedish bank, uses SHA-1 for signing and RSA for encryption.

2.4 Security Tokens

In section 2.2 we saw how OTPs are constructed and used. The user was authenticated based on something he possesed, namely the code scratch card. We can further enhance the security by adding one more factor: We can require the user to know something, for example a PIN-code. This two-factor authentication makes it more difficult to gain access to an account, as a potential attacker would have to steal the card and extort the PIN from the user. Such schemes are implemented by using a security token which requires a PIN code in order to operate. A security token is usually a tamper-proof smart card with a cryptographic processor. Some examples of security tokens can be seen in Figure 5.
Table 2:

<table>
<thead>
<tr>
<th>Step</th>
<th>Action</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>Alice sends Hello.</td>
</tr>
<tr>
<td>2</td>
<td>Bank sends Hello.</td>
</tr>
<tr>
<td>3</td>
<td>The bank sends its Public Key (PK), which is signed by a certificate authority.</td>
</tr>
<tr>
<td>4</td>
<td>Alice verifies the received PK by checking whether it is signed by the certificate authority.</td>
</tr>
<tr>
<td>5</td>
<td>Alice generates a Master key (MS).</td>
</tr>
<tr>
<td>6</td>
<td>Alice encrypts the MS with the bank’s Public key.</td>
</tr>
<tr>
<td>7</td>
<td>Alice sends the encrypted key.</td>
</tr>
<tr>
<td>8</td>
<td>The bank decrypts and obtains the Master Key.</td>
</tr>
<tr>
<td>9</td>
<td>Both Alice and the bank calculate two keys K1 and K2, that are used to encrypt and authenticate the messages.</td>
</tr>
</tbody>
</table>

![Figure 4: SSL connection setup.](image-url)
3 Implementations

3.1 Chip Authentication Program (CAP)

CAP is a relatively new protocol based on the older EMV\textsuperscript{1} standard. It was developed by MasterCard and is based on digitally signing transactions [6]. The advantage of this is that a TAN can be generated precisely for one specific transaction and theft of TANs does not lead to a compromise of the account. Users of this system are issued a smart card, digitally signed by the issuer, containing a cryptographic processor and their private keys. Along with the card they are issued a card reader.

CAP can operate in three modes: identify, respond and sign. Which mode is used and in what way is entirely open to the bank issuing the card.

Every mode will begin with the reader reading the data contents of the card and requesting a PIN from the user to authenticate him. In the next step the cryptoprocessor will create a response based on the entered PIN, an internal 16 bit-transaction counter, the private key and, if present, additional information entered by the user. Note that including the transaction counter makes sure that no response code is ever used twice. In case of the identify-mode, the user will not have to specify anything else. This response code can be used to authenticate a user when logging into the bank website.

If the response-mode is chosen, the bank will send a challenge to the user, which will be included in the computation of the response. This means that the user authenticates a precise action, before the bank executes the

\textsuperscript{1}For more details see [7]
corresponding transaction. See figure 6 for an illustration. 

Sign-mode finally takes two arguments: An account number and a value. The user therefore has to acknowledge a precise transaction, as he has to enter both the target account and the amount of money to be transferred, which significantly hampers phishing attacks.

It would be interesting to see which cryptographic algorithms and parameters are used, but CAP is a proprietary standard and no precise information has been released, safe for some reverse-engineering attempts, which cast a not all too bright light on CAP, as some design parameters might make it susceptible to man-in-the-middle attacks eg. using fake terminals [5].

3.2 RSA SecurID

This scheme basically works very similar to the identify-mode of CAP. However these tokens do not require any card reader, as they have their own displays and keypads. One major difference is that security is strengthened by the fact that the SecurID tokens have a very precise internal clock which is synchronized to a master server against which the users can authenticate themselves.

The 6 to 8-digit response of the SecurID tokens is computed over the PIN, the present time and a 128 bit key, which is unique to every token, using a variant of the AES algorithm. The response generated is only valid for 30 to 60 seconds depending on the model [9]. The SecurID tokens are not very common for online banking usage, because they are rather expensive and are only mentioned to illustrate the idea of having time as a source of entropy and guaranteeing freshness without a challenge/response-protocol.

3.3 Open Authentication (OATH)

The open authentication initiative is an attempt at developing an open standard for 2-factor authentication which should provide means for federated authentication systems like OpenID. The final result should be a holistic, strong and inexpensive cryptosystem, which integrates SIM cards, X.509 certificates and OTPs.

The core of OATH is the HOTP-algorithm [10], which provides the OTP component. HOTP stands for HMAC-based One-Time Password. This algorithm creates a response of at last 6 characters from a shared key of typically 160 bit length and an 8 byte counter which must be synchronized to the server. The key and counter-value will be hashed using SHA-1\(^2\) and the re-

\(^2\)For more details see [4]
sulting 20 byte string is truncated and converted back to decimal to obtain
the HOTP-value to display on the token.

The clients increments the counter with every generation of a value, while
the server only incrementes its counter after successful authentications. This
requires resynchronization. The server will maintain a lookahead window and
check wheather the received value is among those. If yes, the server may ask
for several HOTP-values to ensure that the synchronization is valid and not
fraudulent. It is suggested that the synchrony of the counters could be used
to provide for a two-way authentication: The token can compute a HOTP-
value, which is checked by the server. If the server finds it to be correct, it
can compute the next HOTP-value and relay it to the token, which can now
check this value itself.

4 Conclusion

We have looked at various basic principles of securing online banking and
several implementations, some widely used, others more or less still in devel-
opment. We found that the field is largely governed by proprietary solutions
and it is therefore difficult to find precise technical data for most systems.
It is quite clear that OTPs are here to stay as they significantly mitigate
the dangers of phishing attacks at only slightly increased cost in resources.
2-factor authentication will also remain prevalent as it is a good combination
of protection schemes to have both, something tangible providing high cryp-
tographic strength and something intangible that cannot be easily stolen.

Until the advent of cheaper, more universal hardware systems, such as
suggested by OATH, the combination of PIN/TAN and token generators will
probably prevail, as it provides a good minimum between financial expsenses
for the banks and inconvenience for the users with a strong trend towards
using digital signatures instead of shared secrets. The only probable alter-
native at this point might be distribution via SMS which would be cheaper
for banks, but increase the risk for the customers, as phones are more easily
lost or manipulated, than token devices.
Figure 6: Response-mode of the CAP-protocol. Taken from [5]
References

[1] APACS: Online banking usage amongst over 55s up fourfold in five years
   http://www.apacs.org.uk/media_centre/press/08_24_07.html

[2] APACS: APACS announces latest fraud figures
   http://www.apacs.org.uk/APACSannounceslatestfraudfigures.htm

[3] Reinhard Wobst
   Cryptology Unlocked

   Cryptography - Theory and Practice, p. 137

[5] Saar Drimer, Steven J. Murdoch, and Ross Anderson
   Optimised to Fail: Card Readers for Online Banking


[7] EMV specifications 4.2


[10] RFC 4226
    http://www.ietf.org/rfc/rfc4226.txt