Vulnerability Analysis of the Brazilian Payment System

Dimitri do B. DeFigueiredo
defigueiredo@ucdavis.edu
Department of Computer Science
University of California, Davis

Abstract

In this paper we analyze the security of the new Brazilian Payment System. No worrying vulnerabilities were found, but there are some issues that we believe need to be addressed more carefully. Some minor changes and directions for further development of the system are proposed, such as: specifying which random number generators can be used in the system and encrypting digital signatures to avoid exhaustive search.

Background

On June the 30th of 1999, a board of directors meeting of the Brazilian Central Bank decided to restructure the operation of the Brazilian Financial System [6]. Since then, the system has been completely overhauled. In the payment system in operation until then, the Brazilian Central Bank had to sometimes give the green light to trade transactions even when the financial institution did not have the necessary funds on reserve. This was necessary because some settlement mechanisms were only carried out after day trading was over. The transactions were given a green light in the hope that future same day transactions would restore the necessary reserves to the financial institution. In order to allow for the operation of the system, the Brazilian Central Bank was forced to risk up to R$ 6 billion (approx. US$ 2 billion[1]) of taxpayer's money, in the hope that the financial institutions would redeem their positions at the end of a day's trade. To avoid this problem, a few measures have been taken since:

1. First, each bank's reserve is now monitored on a real-time basis and is not allowed to become negative at any time. For this to become operational, financial institutions must be able to redeem their positions instantly to avoid cash flow problems. Thus, the second measure was taken.
2. A Large-Value Transfer System (LVTS) is now in place to transfer large monetary values. This system operates through Real-Time Gross Settlement (RTGS) to allow financial institutions to redeem their positions instantly.
3. Clearing houses that operate through Deferred Net Settlement have now to install mechanisms to allow for the completion of a day's trade even if the Central Bank refuses payment due to lack of the necessary funds. In order words, the clearing houses should provide certainty of settlement.

Because of the real-time nature of the RTGS system a new communications infrastructure was required. From a technical point of view, the new Brazilian Payment System (SPB) consists of a private communications network where all players of the financial system exchange messages to carry out their daily business. The requirements for the system were laid out in a restructuring plan [3] that clearly indicates reliability and security as concerns. Another requirement for the operation of a RTGS system is the finality of the transactions. Requested payments should be irrevocable and unconditional. This requirement implies that the system should provide a high degree of reliability and availability. The security architecture of the system should address these two issues.

After a transition period, the whole Brazilian financial system now operates through this network for transactions of large value. The amount of money that goes through the network is huge, security needs to be an utmost concern.

[1] values for July 2002.

System Design

It is clear from the requirements laid out for the networking infrastructure that great care has been taken to ensure the reliability of the system, especially through fault-tolerance. The communications network for SPB, called RSFN[2], is an IP based network where each financial institution operates as an autonomous system running BGP on the routers placed in its facilities. According to the specifications [4], RSFN must be isolated from the financial institution's intranet through a firewall. The basic networking layout at a financial institution is shown in fig. 1.

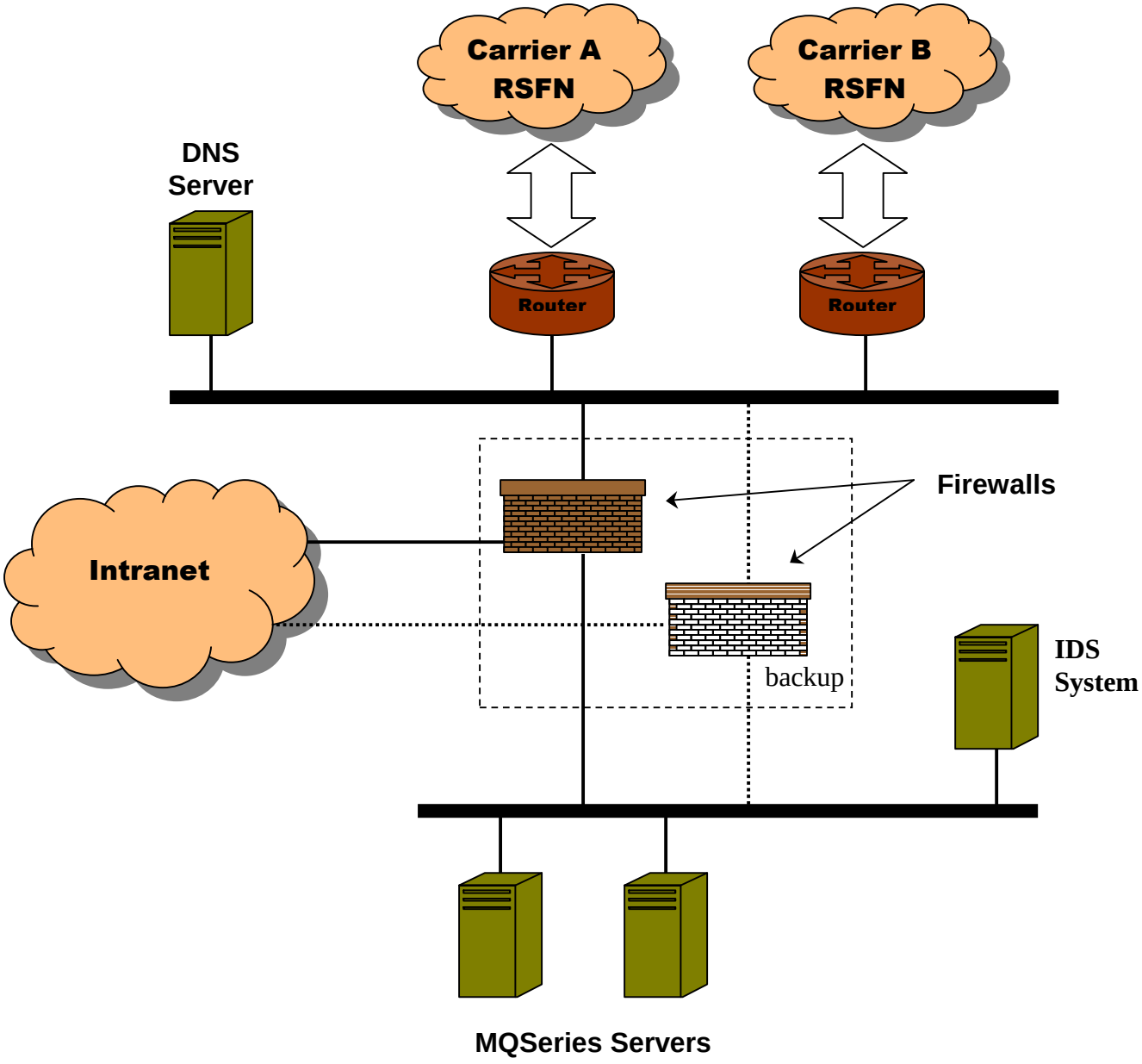

Figure 1. Basic network topology at financial institution (based on [4]).

[2] From the Portuguese: **R**ede do **S**istema **F**inanceiro **N**acional

One of the design goals of RSFN is that it should be able to provide other services in the future [4]. However, this causes the usual tension between security and functionality. Opening the network to other uses increases the number of possible windows that could be exploited. We suggest that such functionalities be made available only when the security and reliability services provided by the network are required. Services such as: confidential conference calls and transmission of critical business proposals in electronic format, are typical of the kind of services we think the network should be used for. But we also believe that the use of the network for other purposes should be considered carefully.

One of the consequences of this design goal is that the system is connected to the intranet of the participating financial institutions. Therefore, it is also open to attacks coming from these networks. The firewall is the gateway between the intranet and the RSFN network and should have some redundancy (backup) to avoid a single point of failure, as pointed out in the specifications [4] and can be seen from fig. 1 above. Yet, one should note that they are also placed between the end severs (MQSeries Servers) and the rest of RSFN. Thus, a compromised firewall would have the capability of intercepting *all* the messages received and sent by the end servers. A topological configuration where the intranet can access RSFN from behind the end servers would provide an extra level of security. Such a solution, although more costly, does not allow for the execution of active attacks (such as the *man-in-the-middle* attack) through a single point of failure.

Financial institutions *must* use a DNS server in the network, this enables the operation of fail-over backup plans. The DNS protocol is well known for its lack of security, perhaps a more secure version should be considered (Notice that the DNS server is open to attacks coming from the RSFN "clouds" in fig. 1).

Security – The Communications Protocol

There is no formal description of the upper level communication protocol used in the Brazilian Payment System. However, a description of how a large number of business transactions should be implemented using the system can be found in the data flow diagrams in [1]. The protocol simply shadows all the different business transactions that may be performed. It is important to point out that all parties operate autonomously and concurrently.

For each real-time business transaction that is to take place, the protocol should provide *certainty of settlement*. Ideally, every transaction once performed should be final, *i.e.*: fraud-proof, indisputable, undeniable and accountable. Furthermore, the business transactions should also remain confidential. To ensure such characteristics, a set of security measures has been adopted for each message transmitted in the system. These aim to achieve authenticity, non-repudiation and confidentiality for each message [2] and thus provide the same goals for each transaction as a whole. Naturally, one needs to be concerned about whether this gap between the message level and the transaction level can be exploited, especially when dealing with confidentiality. If traffic analysis can be performed precious information could leak. For example, even though it should be no major security problem, simply by examining the size of the authenticated part of all messages, it is possible to determine with high accuracy when a message is the echo message (GEN0001). These messages are smaller than all the other messages, even considering padding. Thus, one could argue that it is not useful to encrypt theses messages unless random padding is used. On the other hand, decrypting these messages requires a very large amount of computational resources per message as asymmetric encryption is used once per message received.

Availability is only considered at the network level (RSFN) and is not addressed in the communications protocol. Quite the opposite, servers are required to implement functionality that may render them more vulnerable to Denial-of-Service attacks. Specifically, financial institutions are required to respond to echo messages and to log all incoming and outgoing messages. These requirements make perfect sense when considering accountability and reliability, but they may make it easier to compromise system availability. A replay attack that uses traffic analysis to identify and record an echo message and then floods the recipient with those messages can easily overwhelm the computational resources of an authenticating/decrypting server. We believe that liability should be addressed clearly. A "malfunctioning" server flooding another institution and preventing it from operating should be accountable for its actions.

The Message Format

At the Application Layer, the message format used in SPB is made up of 5 basic blocks as shown in fig. 2.

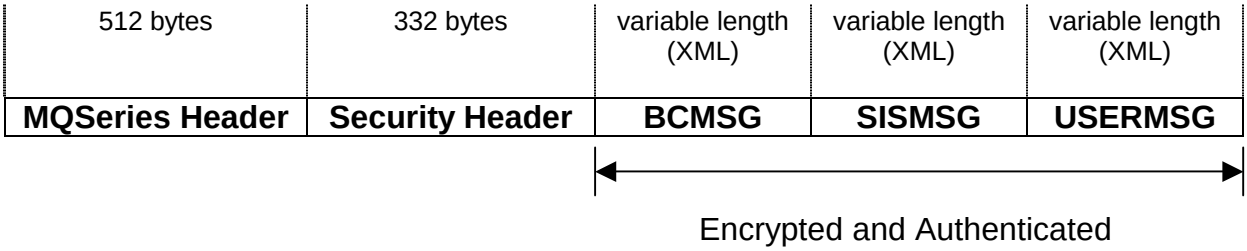

Figure 2. Layout of the message format in SPB at the Application Layer.

The first block is the header for the messaging software used (MQSeries). This part has a fixed length and is not authenticated or encrypted. The next block is the security header in binary format, it is 332 bytes long. It contains all the necessary information to encrypt and authenticate the remaining parts of the message. The last 3 blocks called BCMSG, SISMSG and USERMSG (control, system and user blocks) are coded in XML using UNICODE. All 3 blocks have variable length, as one would expect in XML, but the first 2 have strict formatting rules. These blocks are authenticated and encrypted before transmission.

Fig. 3 below shows the structure of the security header. Messages are signed using a cryptographic hash function (either MD5 or SHA-1) and RSA; and then encrypted, using Triple-DES in CBC mode (so far these are the only options that can be used). Only the XML part of the messages undergoes such processing. The digital signature found in the security header is NOT encrypted.

field number	MQSeries Header	
1.	Security header length	2 bytes
2.	Version	1 byte
3.	Error code	1 byte
4.	Special Processing Flag	1 byte
5.	Reserved	1 byte
6.	Asymmetric Encryption Protocol	1 byte
7.	Symmetric Encryption Protocol	1 byte
8.	Digital Signature Protocol	1 byte
9.	Cryptographic Hash	1 byte
10.	Recipient's Certification Authority ID	1 byte
11.	Serial Number of Recipient's Certificate (Encryption)	32 bytes
12.	Sender's Certification Authority ID	1 byte
13.	Serial Number of Sender's Certificate (Signature)	32 bytes
14.	Encrypted Symmetric Key	128 bytes
15.	Digital Signature	128 bytes
	Encrypted contents	

Figure 3. Security Header Layout.

The security header format is quite concise. The first field indicates the total header length and is fixed at 332 bytes for the first version of the protocol. The second field indicates what protocol version is being used. These two fields along with the "Reserved" field are used for compatibility with future formats. The Error Code field is left at zero when no error occurred and is used when sending an erroneous message back to its source. The Special Processing Flag is used to signal some rare conditions that require extra processing by the recipient. For example, if the contents are compressed and therefore not in XML format or if the certificate being used for encryption is new. The following two fields specify the encryption schemes used. The only options available for fields 6 and 7 in version 1 of the format are RSA and Triple-DES respectively. Each message is encrypted using the usual hybrid encryption scheme: A session key is selected at random and used to encrypt the contents using the symmetric algorithm. Then, the symmetric key itself is encrypted using the recipient's public-key. To ensure the correct public-key is used, it is obtained from the certificate specified in fields number 10 and 11. To sign the message, the sender hashes *the contents following the security header* (padding with zeros if necessary) using the hash function specified in field 9 and then signs the hash using the protocol specified in field 8. Currently, RSA is the only digital signature protocol allowed. The signature is then stored *unencrypted* in the last field. The security header itself is not authenticated.

The documentation available does not specify the algorithm used to randomly select the symmetric key. Even though it is reasonable to assume that most parties will implement the system with adequately chosen algorithms, perhaps a few guidelines on what would be acceptable random number generators (*e.g.* ANSI X9.17) should be proposed upfront.

The fact that the digital signature is not encrypted seriously undermines the confidentiality provided by Triple-DES. An attacker with some a priori knowledge of the message transmitted could use this information to circumvent the encryption scheme. For example, suppose that through the size of the message and some other traffic analysis and side information the attacker knows that the message being transmitted is a funds transfer between two financial institutions (STR0004). The attacker's goal is to know the total amount transferred. The standard formatting for the XML part of any SPB message looks like the following:

```
<?xml version="1.00"?>
<!DOCTYPE SPBDOC SYSTEM "SPBDOC.DTD">
<SPBDOC>
  <BCMSG>
    <Grupo_EmissorMsg>
      <TpIdentdEmissor> sender's id type ("P" in this version) </TpIdentdEmissor>
      <IdentdEmissor> sender's unique ID </IdentdEmissor>
    </Grupo_EmissorMsg>
    <Grupo_DestinatarioMsg>
      <TpIdentdDestinatario> recipient's id type ("P") </TpIdentdDestinatario>
      <IdentdDestinatario> Recipient's unique ID </IdentdDestinatario>
    </Grupo_DestinatarioMsg>
    <Grupo_Seq>
      <NumSeq> Sequence Number </NumSeq>
      <IndrCont> Segmentation flag </IndrCont>
    </Grupo_seq>
      <NUOp> Unique Operation Identifier (sequential) </NUOp>
  </BCMSG>
  <SISMSG>
      ... This is the system block (in this example it should be formatted as defined for STR0004)
  </SISMSG>
  <USERMSG>
      ... This is the user block
  </USERMSG>
</SPBDOC>
```

Figure 4. The basic XML layout for SPB messages.

Taking a closer look at the fields in the control block one can observe that they are either uniquely determined by the sender and recipient or they are of sequential nature. The same reasoning applies to the fields present in the system block of STR0004 messages. There is no underlying randomness to the message other than that provided by the encryption algorithm itself. In other words, the message space is fairly small. The attacker can simply generate all the possible messages and verify the signature until a match is found. This requires some computational resources, but could be extremely successful depending on how much side information is available. Obviously, simply encrypting the signature would thwart such attacks. If such approach is taken and the signature is encrypted at the end of the message both hashing and encryption (in CBC mode), which have to be done sequentially, can still be performed almost simultaneously and no serious performance penalty is introduced.

A very important part of the control block in all messages is the *recipient's ID*. The proposed scheme provides both authentication and encryption. In the standard, messages are signed before they are encrypted. This is space efficient, the ciphertext does not have to be kept for later use as part of the authenticator. However, it does not bind messages to the recipient. If these were the only components of the messages, anyone who received an authenticated message could retransmit it. For example, if bank A sends to bank B an authenticated message. Then, bank B can impersonate A by retransmitting the same message to C. This could impair the use of authenticated messages as evidence in a Court of Law as A can send a message to B and later claim that the message was intended for someone else. A defense against this kind of fraud is to include an identification of the desired recipient in the body of the authenticated message, this is achieved by the *recipient's ID* field in the control block. However, when transmitting files over the network, the control block may not be transmitted and thus the system is open to such attacks. The *Unique Operation Identifier* prevents replay attacks from an untrustworthy recipient. It is also important to force the implementing software to check that messages received are indeed addressed to the right recipient. This should probably be part of the certification tests[3]. Perhaps, a more explicit description of the security role played by the two control block fields just mentioned should be included in the documentation.

System Level Issues

A few security characteristics of SPB should be considered together with the environment in which the system operates. A great deal of detail has been provided in the documentation about the certification process. However, one issue that the certification process fails to address is that because the value of the transactions in the system may be very large, liability for misbehavior cannot always be enforced. A certification authority with a liability of R$ 5 million may certify the authenticity of transactions in excess of R$ 500 million. For example, new certificates are only signed by, the certification authority and the new private-key. Using the new private-key and not the old one ensures that a compromised private-key cannot be used to hijack a *key chain* for a long period of time, but it gives a lot of power to the authorized certification authorities. Thus, if a certification authority decides to produce fake certificates for most, *if not all*, of the participating financial institutions, it is able to do so. Furthermore, if it is then able to make those certificates available, it may be able to perform fraudulent transactions of large value and secure funds overseas in large excess of its liability. Perhaps other mechanisms enforced by the Central Bank itself should be considered before updating keys. This need not be very extensive, but should give the Central Bank some control over this problem.

The accountability of system rests on the log files produced by all the participating financial institutions. These log files are much more useful if they can constitute reliable legal evidence that can be made available in case of future disputes, so the usefulness of the log files is undermined by the threat of late modification. Ideally, all information in the log file should be authenticated. This is certainly not the case in SPB, where the security header itself and the date and time of each message are recorded in unauthenticated form[4]. Perhaps, a more costly but resilient solution should be sought.

A final issue that was rightfully left to the auspices of each financial institutions is the enormous gap that there exists between use of a public-key system by computer software and the appropriate

[3] Unfortunately, no such information was available and thus the focus of this work is on the design level specifications of the protocol. The implementation is not studied directly.

[4] The information can be correlated to other sources but cannot be made as precise as would be possible if it were included in the authentication process.

authorization by bank personnel, such as the Chief Operations Officer. Each financial institution is responsible for its own private-keys [2]. How their security policy should achieve this is an open question. One of the few times this is addressed in the documentation is in the recommendation for financial institutions to acquire special hardware to keep their private-keys safe. Certainly, some security policy guidelines, like the Principle of Separation of Duty, should be adopted by each financial institution. Whether these should be made mandatory is debatable, but a few recommendations would not hurt.

Conclusion

This work has performed a security analysis of the Brazilian Payment System at the design level. The system is based on well-known cryptographic primitives and did not present any major concerns. The study has found that a few important points should be made more explicit in the documentation to avoid generating problems at the implementation level. Namely, it is our belief that the documentation should determine which Random Number Generators are acceptable for use in the system. That there should be a mechanism to determine the intended recipient of authenticated files sent over the network, that the digital signatures should be encrypted. And finally, that the system level issue of the liability of the certification authorities could be addressed in future revisions.

All in all, we believe that the security characteristics of the system strike a very good balance between cost and functionality.

Bibliography

All material is available on-line (in Portuguese) at: http://www.bcb.gov.br

[1] “Catálogo de Mensagens”, Sistema de Pagamentos Brasileiro, volumes 1 and 2, version 1.01. Brasília, Brazil, 30th Nov 2001.

[2] L. G. Guimarães, F. Burgos, “Manual de Segurança de Mensagens do SPB”,GT-Segurança, version 2.1 April 2002.

[3] "Projeto de Reestruturacão do Sistema de Pagamentos Brasileiro", Nota Técnica, Banco Central do Brasil, Brasília, Brazil, 31 Oct 2000.

[4] “Rede do Sistema Financeiro Nacional”, Manual Técnico, SPB/ RSFN, version 3.0, Nov 2001.

[5] “Reestruturação do Sistema de Pagamentos Brasileiro, Conceitos e Considerações” (available at http://www.bcb.gov.br/ftp/deban/deban-pdf.pdf)

[6] “Reestruturação do Sistema de Pagamentos Brasileiro, Diretrizes do Projeto de Reestruturação” (available at http://www.bcb.gov.br/htms/infsispag.htm)

[7] “Reestruturação do Sistema de Pagamentos Brasileiro, Ententendo a Reestruturação” (available at http://www.bcb.gov.br/htms/spb/entendendoreestruturacao.shtml)