A&QT-R 2002 (THETA 13)
International Conference on Automation, Quality and Testing, Robotics
May 23-25, 2002, Cluj-Napoca, Romania

SECURE E-MAIL SYSTEM

Eng. Serban Badila, Pro3Soft Cluj-Napoca
Eng. Mihai Chezan, Technical University Cluj-Napoca
Eng. Cristian Madularu, Technical University Cluj-Napoca
Eng. Sorin Chiorean, Technical University Cluj-Napoca
Eng. Silvano Sanchez, SATA HTS Italy
Eng. Federico Dalle Mese, SATA HTS Italy
Prof.dr.eng. Gavril Toderean, Technical University Cluj-Napoca

Email: toderean@cluj.astral.ro
Web: http://pages.astral.ro/pro3soft

ABSTRACT

The SATA Secure E-mail System, as its name says, is a mail system that was designed from the ground-up with security considered as the number one priority. Because currently mail protocols do not provide strong built-in security specifications, it was mandatory for us to develop a secure-mail protocol. The developed system proved to be both safe and user-friendly.

The presented software is developed for, and in collaboration with, the Italian SATA Consulting company.

KEYWORDS: Secure E-Mail System

1. INTRODUCTION TO THE SECURE E-MAIL SYSTEM

The SATA Secure E-Mail system is built as a client-server model. The client was developed in Visual Basic because of this language's capabilities in the area of developing easy, nice and maintainable user interfaces. The server was developed in Visual C++ 6.0 with MFC support, a well suited language for server-side development [2], [3].

The system works in a classical client-server fashion, communication being implemented over the TCP/IP protocol. While the server is running and listening for incoming connections, the client ,which is located on a remote computer, can connect to the server. After a successful authentication, the client can request different tasks for the server to process: read mail or documents, send mail or documents. All these tasks are processed using a security system which will be exposed in this paper.

2. DESCRIPTION OF THE SECURE E-MAIL SYSTEM

While a large application development process can really succeed only if it is split in smaller components, i.e. modules, the same thing applies to the documentation of an application. Therefore, the description of the system will be split in three parts: the security model, the client and the server. The server description will also be split in several parts.

The security model:

In a short description, the secure transmission is implemented by encrypting data using an 128 bit key. The key is always changed when a new transaction starts. This model is somehow designed in the same manner as military systems [4].

The messages and documents are also kept encrypted on the server, thus no one, besides the owner, is allowed to have clear access to them.

The client:

Visual Basic behaved very well regarding user interface programming. From the user's point of view, it is yet another nice and easy-to-use Windows application, while from the programmer's point of view we have a clean code allowing easy maintenance and extensibility.

The client application has a lot of features and because of its intuitive look & feel the client will feel comfortable while using it.

The user can perform the following operations:

- send/receive messages with attachments for multiple recipients
- store documents on the server in a secure way; this means that the original file from the local computer is sent to the server, its content is over-written with null characters and then is erased. This way, the file can never be recovered from the client's computer.

The server:

<u>- General concepts:</u>

The complexity of the server proved that it is mandatory to use a very versatile and full-featured programming language. That is why we chose Visual C++ 6.0.

Here is how a client-server cycle works [1]:

1. After it is started, the server opens a socket and binds it to a dedicated well-known port, and starts listening for incoming connections from clients.
2. When a client connects, the server asks the client to authenticate, using a user/password which is matched over the real user/password kept encrypted in a database described below.
3. After a successful authentication process, the server tries to perform any operation requested by the client, such as read messages or documents, send messages or documents.

4. The server disconnects the client either when the client requests it, or after a specified timeout interval elapses.

- The database model:

The database is composed of five tables as described below:

- a table that contains users personal information such as name, password, services that the user can access, space quota, etc.
- a table that contains the security keys involved in the client-server communication
- a table with references to the received messages
- a table with references to the sent messages
- and a final table with references to the stored documents.

For security reasons, there is always a backup of the entire database, which is used in case that the primary database is corrupted [5].

-The monitor and log files:

In any client-server system it is a good practice to keep information on the server regarding clients operations. This way, the system is more maintainable and the security of the system is also increased because of the possibility of observing unusual operations. For example, when a client reports that he cannot find a document stored in his folder, by checking the logs someone can establish if it was a server error or someone really erased the document that could not be found.

The clients operations can be checked in real-time using the monitor feature. This feature is started along with the server and it displays real-time information about the connected users operations. This information is updated at a specified timer, usually one second.

All clients operations are stored in log files for later checking. For ease of information searching, a new log file is created each day.

- Future enhancements:

In order to extend the usability we plan to enhance the server by adding SMTP (Simple Mail Transport Protocol) support. This way, the user can also deal (receive/send) with normal mail, eliminating the need to use another mail system to talk with usual SMTP servers (Yahoo Mail, Hotmail, etc). Of course, this means that the messages will be sent/received in an unsecured way when talking to other SMTP servers, but they will be stored encrypted in the user's mailbox.

3. CONCLUSIONS

The Secure E-Mail System described above was designed to be used in corporations (organizations, communities) where security is a must. While the server resulted in an industrial-strength architecture with strong scalability and availability

skills, the client's user interface is simple and intuitive. New users, even new comers in the computers field, will feel comfortable using it.

BIBLIOGRAPHY

[1] SATA Secure E-mail System Protocol Specification
[2] Prosise J., 1999 Microsoft Press, "Programming Windows with MFC"
[3] Microsoft Platform SDK, MSDN Library
[4] CryptoAPI v2.0, Microsoft Platform SDK, MSDN Library
[5] ODBC SDK Programmer's Reference, MSDN Library