SECURE E-MAIL APPLICATION SOFTWARE FOR GOVERNMENT IN INDONESIA

Kridanto SURENDRO and Setiyo CAHYONO

Abstract: Exchanging information using e-mail brings a good deal of vulnerability that can be exploited by an unauthorized third party for individual or organizational purposes. This is quite probable since e-mail systems are designed to provide a straightforward and fast way of information delivery without considering the security of information. Prior to applying any specific security solution, an organization has to consider system characteristics and the existing problems through evaluation of security needs and faced risks. An approach that can be used to determine the security needs of an organization is risk management. Risk analysis can aid the organization in identifying the risks, why there can be a risk, to determine priorities and create prevention strategy to reduce the risks. In this article, the authors discuss the development of secure e-mail software. E-mail protection is accomplished using Secure Socket Layer (SSL) to protect the communication between the web server and the local computer, encrypting e-mail messages with combination of public and symmetric key encryption, dynamic encryption key and adding a digital signature. The experimental results show that the software can be used to protect information exchange and can reduce such security threats as eavesdropping, identity theft, false message, message modification and repudiation. Using encryption expands the size of the e-mail message to 161.96% from the actual size and the time required for encryption process is increased with 3.68%.

Keywords: Risk Analysis, Cryptography, Secure E-Mail, e-Government.

E-Government is being developed in Indonesia as a means to provide electronic-based services to citizens, to improve the quality of public services in an efficient and effective manner. Indonesian government has issued national policy and strategy for e-Government development[1] through President Instruction No.3 from 2003 as a foundation and a framework for the whole process of e-Government development. However, a consistent and supportive regulation, a standard, and a guidance are still needed, to conduct the e-Government development in a systematic and an integrated way.

E-mail is one component of the e-government services that will be implemented in local and central government organizations. Therefore, the information sent through e-mails has to be secure; it could contain confidential and urgent information. The implementation of security mechanisms in the e-mail system will reduce the risk an unauthorized party to use the information improperly for individual, group, or even one nation to another interests.

This article will discuss several aspects that can be used in determining e-mail security requirement standard for the Indonesian government. The design of an e-mail application that can assure and secure data exchange will also be presented by the authors. Hopefully, the results will become data transfer and exchange standard application for the government offices.

Basic Concept of Cryptography

Cryptography[2] is a branch of cryptology (science or study about cryptography) that deals with algorithmic design for encryption and decryption, ensuring privacy and authentication of a message. Encryption is a transformation process from a plain text (clear text and data) to some meaningless forms (cipher texts). Decryption is a reverse process to encryption. Encryption algorithms are employed for the encryption and decryption processes. The resource used for encryption and decryption is called a crypto-system.

The general encryption and decryption operations can be described as follows[3]:

$$Y = E_{KE}(X) \qquad \text{(encryption)}$$

$$X = D_{KD}(Y) \qquad \text{(decryption)}$$

where X is plain text, Y is cipher text, KE and KD are encryption and decryption keys, respectively. A general crypto system is illustrated in Figure 1.

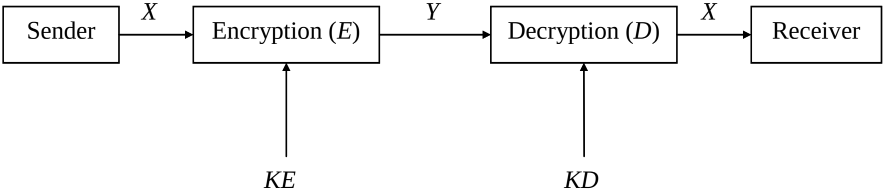

Figure 1: General Concept of a Cryptosystem.[4]

Data Encryption Standard

Data Encryption Standard (DES)[5] is a block-type cryptographic key algorithm. It takes 56 bits length key. The key is usually a 64 bits number, where every 8-th bit is used for a parity bit. Basic building block for DES is a combination of a substitution technique followed by permutation of the text, according to the key. This is known as a *round,* and DES consists of 16 rounds.

DES operates on 64 bits plain-text blocks. After initial permutation, each block is divided into two parts, a left part and a right part, with 32 bits in each part. Then 16 rounds follow in which similar operations/functions (called *f*) are performed, and data are combined with the key. After the final 16-th round, left and right parts are joined and a final permutation (reverse process to initial permutation) is performed. On each round, the bit key is shifted and 48 bits out of the 56-bits key are chosen. The left parts are expanded to 48 bits by expansion permutation, the shifted 48 bits and the permutated key are combined using XOR, resulting into 8 Sbox new 32 bits, and permutated again. Those four operations form the *f* function. The result after the *f* function is then combined again with the left parts using XOR, to produce new right parts. Old left parts then become new left parts. And these operations are repeated 16 times.

Rivest-Shamir-Adleman (RSA)

Rivest-Shamir-Adleman (RSA) is a public key cryptographic algorithm, invented by Ron Rivest, Adi Shamir, and Leonard Adleman from MIT in 1977. Difficulty to factoring big numbers is the major advantage of RSA. Public and private keys are a pair of big prime number (100-200 digits or even bigger).

A pair of keys can be produced considering the following procedure[6]:

1. Choose randomly two big prime numbers (assume p and q)
2. Calculate $n = p*q$
3. Choose randomly encryption key (e), where e and $(p-1)*(q-1)$ are relatively prime.
4. Calculate $d = e^{-1}\mathrm{mod}((p-1)*(q-1))$.

Numbers e and n are public keys, numbers d and n are private keys. The sender and receiver have to tell the n value. The sender knows the e value, and the receiver knows the d value.

Before encryption, the plain text M is divided into blocks, where each block has a binary value less than n. Algorithmically, encryption and decryption look like:

$$C = M^{e}\,\mathrm{mod}\,n \quad \text{(encryption)}$$

$$M = C^d \bmod n \quad \text{(decryption).}$$

Radix-64 conversion (Base 64)

Radix-64 conversion or Base 64 is used to convert binary input to printable characters. The conversion form has several characteristics, as stated below[7]:

- The range of the conversion is sets of characters that universally represent the whole set, not specific to some character sets. As such, those characters can be converted into a form required by the system. For example, the "E" character can be represented as 45 hexadecimal (in ASCII-based system) and CS (in EBCDIC-based system).
- The character set consists of 65 printable characters; one of them can be used as a padding character. With 2^6=64 characters available, each character can be used to represent a 6-bits input.
- No control characters are included. It means that a message can be converted into radix-64 form by the mail handling system that scans data stream for character control.
- The character "-" is not available. This character has its own meaning in RFC822 format and has to be avoided.

Security Requirements and Risk Analysis

Before implementing a security solution to an organization, security requirements and risks have to be evaluated thoroughly. One approach is to adopt an organization's perspective and identify what needs to be secured, why there could be a risk, and to propose a solution. This can be achieved using a risk management approach. Risk management is a continuous process to identify the risks and to implement a relevant plan to overcome it.[8]

Risk management consists of several activities, including[9]:

- Identification of risk to information security
- Risks analysis and determination of priorities
- Planning for improvement and reducing risks by developing security strategy
- Planning how to implement security strategy and reducing risk by developing complete activity plans. This activity including cost-benefit analysis is based on strategy and activities.
- Implementation of the chosen activity plan
- Monitoring of plan's improvement and effectiveness

- Control of variations in plan execution by conducting corrective action if necessary.

Risk Identification

Threat Identification

Common threat sources can be divided as[10]:

- *Natural Threats.* Floods, earthquakes, tornadoes, landslides, avalanches, electrical storms, and other similar events.
- *Human Threats.* Events that are either enabled by or caused by human beings, such as unintentional acts (inadvertent data entry) or deliberate actions (network-based attacks, malicious software upload, unauthorized access to confidential information).
- *Environmental Threats.* Long-term power failure, pollution, chemicals, liquid leakage.

This article will only discuss human threats, with the assumption that the remaining two threat sources have been reduced by the organization.

Identification of Vulnerability

Vulnerability[11] is a lack or weakness of the system either from security, design, implementation, or internal control procedures that will likely be exploited un/intentionally and will cause security system failure or violation. Mail vulnerabilities can be listed as follows[12]:

- Sending e-mail plainly; Sending e-mail from client to server and vice versa is conducted in plain form.
- Saving e-mail plainly; E-mail messages are saved in plain form to all SMTP servers, backup can be done from server and client computer. Backup of mail server is completed with copy of the message, and saved messages backup can be maintained for a certain period.
- Open port; Use of port 25 for SMTP and 110 for POP might trigger vulnerability to mail server.
- User authentication to mail server plainly; when using e-mail service, user authentication is conducted by sending username and password in plain form to the mail server.

Risk might occur, when existing vulnerability to system/application can be exploited by threat. Table 1 describes risk that is likely to occur when existing vulnerability can be exploited by threat.

Table 1: Vulnerability That Can Be Exploited by Threat.

No	Vulnerability	Threat
1	Sending e-mail plainly	Eavesdropping Message modification False Message Repudiation
2	Saving e-mail plainly	Message modification
3	Open port	Denial of service Spam Virus
4	User authentication to mail server plainly	Identity theft

Risk Analysis

Risk-level matrix has to be obtained in order to measure and scale the risk. Hence, a 3x3 matrix is constructed considering threat (high, medium, low) and impact due to its occurrence (high, medium, low). A qualitative approach is used for measurement. Risk scale is determined using likelihood, impact, the risk-level matrix, and the necessary action according to risk scale.

A survey on e-mail security has been conducted in order to understand the likelihood level of risk due to existence of threat that exploited vulnerability.[13] The results are shown in Table 2 and Table 3.

Table 2: Interview Results of Likelihood Level.

No	Vulnerability	Threat	Likelihood
1	Sending e-mail plainly	Eavesdropping	High
2	Sending e-mail plainly	Message modification	High
3	Sending e-mail plainly	False message	Medium
4	Sending e-mail plainly	Repudiation	Medium
5	Saving e-mail plainly	Message modification	Medium
6	Open port	Spam	Low
7	Open port	Virus	Low
8	Open port	Denial of service	Low
9	User authentication plainly	Identity theft	High

Table 3: Interview Result of Impact Level.

No	Vulnerability	Threat	Impact
1	Sending e-mail plainly	Eavesdropping	High
2	Sending e-mail plainly	Message modification	High
3	Sending e-mail plainly	False message	High
4	Sending e-mail plainly	Repudiation	Medium
5	Saving e-mail plainly	Message modification	High
6	Open port	Spam	Low
7	Open port	Virus	Medium
8	Open port	Denial of service	High
9	User authentication plainly	Identity theft	High

Based on the results of the survey, the risk scale can be measured using risk-level matrix (see Table 4).

Table 4: Risk Scale Measurement Result.

No	Vulnerability	Threat	Likelihood	Impact	Risk Scale
1	Sending e-mail plainly	Eavesdropping	High	High	High
2	Sending e-mail plainly	Message modification	High	High	High
3	Sending e-mail plainly	False message	Medium	High	Medium
4	Sending e-mail plainly	Repudiation	Medium	Medium	Medium
5	Saving e-mail plainly	Message modification	Medium	High	Medium
6	Open port	Spam	Low	Low	Low
7	Open port	Virus	Low	Medium	Low
8	Open port	Denial of service	Low	High	Low
9	User authentication	Identity theft	High	High	High

Security Strategy to Reduce Risk

A strategy to overcome risks can be conducted after risk scale measurement. Security strategy is only implemented to any risks that have high or medium level. The security strategy proposed in this article to reduce risk based on measurement result is described in Table 5.

Table 5: Security Strategy.

No	Vulnerability	Threat	Security	Control type
1	Sending e-mail plainly	Eavesdropping	Encryption of sent and saved e-mail message in storage media	Prevention
2	User authentication plainly	Identity theft	Securing transmission line between client and server using Secure Sockets Layer/ Transport Layer Security (SSL/ TLS)	Prevention
3	Sending/ saving e-mail plainly	Message modification	Adding message digest or finger print to e-mail message	Detection
4	Sending e-mail plainly	False message	Adding digital signature to e-mail message	Detection
5	Sending e-mail plainly	Repudiation	Adding digital signature to e-mail message	Detection

E-mail Security Strategy

Determining the e-Mail Application

The authors choose to implement a web-based e-mail application based on the following criteria:

- E-mail could be accessed from any computer anywhere.
- Users do not need to configure and install any specific e-mail application on their local computer. The application needs to be installed only on the web server.

- Local computer specifications will not pose restrictions to run the application since it locally resides at the server side.
- No restrictions exist about the operating system on the local computer. The application requires only a web browser.

In order to increase the security of a web-based application, security requirements between the local computer and the web server have to be confirmed. Communication security has to be conducted using SSL.[14] An advantage using SSL is that the users can be sure that they have access to the right server. Another is that using SSL in a web-based e-mail will ensure that all the communication between the local computer and the web server is encrypted, thus someone may not easily eavesdrop the original data. SSL will only secure the transmission line between the local computer and the web server, and the data will not be encrypted after being sent to another server.

Secure e-Mail Message Format

There are two formats in an e-mail message: a message header and a message body. To separate the message header from the message body, an empty line is used. The message header contains such information about an e-mail as sender's e-mail address, destination e-mail address, and subject. The message body contains the message itself.

The information contained in the message header is used by the mail server to deliver the e-mail to its destination. It explains why the encryption process can only be applied to the message body.

The encryption and decryption processes would have been much easier if session key, profile-id, digital signature, bit check, date, and time sent were added to the message body. Bit check is the first four characters of fingerprint and is used to check whether decryption to digital signature is successful or not. Fingerprint is created using hash function from sender's e-mail address, time sent, and message content. Figure 2 illustrates the e-mail message format that will be used.

Session Key Generation

Symmetric key cryptography applying the same key for the encryption and decryption processes of a message is used. These keys have to be maintained properly in order to prevent the access of an unauthorized person. To increase security, the key should have the following characteristics: to be random, hard to predict, and to be used only once. A key that is used only for one encryption is called a session key. Session key generation will become a problem if the user has to insert a key each time a message has to be encrypted. The system will handle the creation of session key.

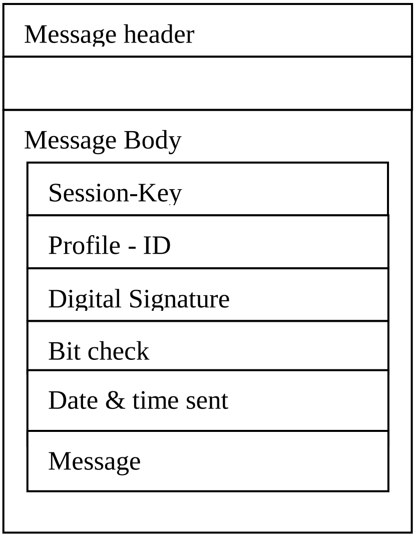

Figure 2: Secure e-Mail Message Format.

The session key will be generated as follows:

- Generate random numbers in the range from 0 to 255; as many as 100 numbers. The random number generator can be facilitated using a function inside the programming language;
- Convert these random numbers into ASCII characters;
- Calculate hash value with function hash MD5 algorithm to produce 32 hexadecimal characters;
- Convert hash value with base64 to produce 44 ASCII characters.

Using the above procedure, the session key will be different each time it is generated. Thus, different encrypted messages will be produced from the same message.

E-mail Compatibility

The e-mail system is limited to allow only printable ASCII characters. The encrypted message could consist of ASCII characters from 0 to 255, hence it cannot be directly transmitted to the e-mail server system. To overcome such a limitation, the encrypted message needs to be converted into printable ASCII character form. Base64 conversion may be used to produce printable ASCII characters. This is enabled due to the fact that the characters in base64 consist of the letters A-Z, a-z, the digits 0-9, “+”, “/”, “=”, with no character control.

Digital Signature Generation

Digital signature is used as a proof that the original sender has composed the e-mail. This is one way to avoid denial of e-mail from both the sender and the receiver.

The process of digital signature generation can be described as follows:

- Create a fingerprint based on date/time of creation and the message using a hash function;
- Create digital signature by encrypting the fingerprint with cryptography public key using sender's private key;
- Add the digital signature to the message.

Generating a fingerprint based on date/time of creation and the message is performed to ensure that the digital signature is only valid for that message and the date/time when the e-mail has been created. Adding a digital signature to the message guarantees that the sender will not be able to deny the message. This is due to the fingerprint of the encrypted with cryptography public key message that highly depends on private key. On the other hand, the private key belongs only to the sender.

Encryption and Decryption Processes

A specific algorithm (whether encryption, a hash function, or a digital signature) will not be applied in the development of secure e-mail. This will provide flexibility and it will enable the use or addition of a specific algorithm. Encryption algorithms used for encryption, a hash function, and a digital signature are shown in Table 6.

Table 6: Cryptography Public Key, Symmetric Hash Function, and Digital Signature Algorithm.

No	Algorithm name	Type
1	Rivest Shamir Adelman (RSA)	Cryptography public key Digital signature
2	Data Encryption Standard	Cryptography Symmetric key
3	Triple DES	Cryptography Symmetric key
4	Rijndael	Cryptography Symmetric key
5	Blowfish	Cryptography Symmetric key
6	Message Digest 5 (MD5)	Hash function
7	Secure Hash Algorithm (SHA)	Hash function

The e-mail encryption process can be conducted as follows:

1. Generate a session key.

2. Generate a fingerprint from date/time of creation and message content using a hash function.
3. Generate a check bit by taking the first four characters from the fingerprint.
4. Generate a digital signature by encrypting the fingerprint with cryptographic public key using sender’s private key.
5. Encrypting the check bit, date/time of creation and message content with a cryptographic symmetric key using the session key.
6. Encrypting sender’s profile-id and the session key with a cryptographic symmetric key using sender’s public key.
7. Combining the encrypted session key, the digital signature, and the encrypted message, and then converting it using base64.
8. To distinguish the encrypted e-mail, an identifier tag is added, “----BEGIN SECURE E-MAIL MESSAGE----” at the beginning of the message body and “----END SECURE E-MAIL MESSAGE----” at the end of the message body.

The e-mail decryption process can be conducted as follows:

1. If the message body begins with “----BEGIN SECURE E-MAIL MESSAGE----,” then it is assumed that the e-mail is encrypted.
2. Extract the message body from “----BEGIN SECURE E-MAIL MESSAGE----” to “----END SECURE E-MAIL MESSAGE----”
3. Convert the message body into ASCII format using base64.
4. Split the message body into encrypted session key, digital signature, and encrypted message.
5. Decrypt the encrypted session key with cryptographic public key using the receiver’s private key to obtain the session key and sender’s profile-id.
6. Decrypt the encrypted message with cryptographic symmetric key using the session key to obtain the check bit, date/time of creation and message.
7. Create a fingerprint from date/time of creation and message using a hash function.
8. Decrypt the digital signature with cryptographic public key using sender’s public key.
9. Compare the first four characters from the decrypted fingerprint, the digital signature, and the check bit. If they coincide, then the decryption to a digital signature has been successful.
10. Compare the fingerprint resulting from the 7-th and 8-th processes. If they coincide, then it is said to be a valid fingerprint.

E-mail Security Architecture

All processes required for e-mail management, including the encryption and decryption processes, are conducted at the server side. The web server where the secure e-mail prototype is installed is completed with SSL. If the user accesses uniform resource locator from the prototype, then SSL will be automatically activated to communicate with the web server by the browser application. In order to obtain maximum security, all the communication between the local computer, the web server, and the mail server has to be accomplished using SSL. However, in this article, SSL only applies to secure the communication between the web server and the local computer.

The process of retrieving/reading e-mail is:

1. The receiver logs on to the secure e-mail application.
2. The web server accesses the receiver's mail server, downloading e-mail and saving the messages into a database.
3. To read the e-mail, the web server retrieves the e-mail from the database.
4. If an encrypted e-mail is found, then the web server will first perform decryption, and subsequently will pass the result to receiver's computer.

The process of sending e-mail looks like:

1. The sender logs on to the secure e-mail application.
2. The sender composes a new e-mail massage and sends it to the web server.
3. If the destination address is registered into the database, then the web server will first perform encryption. On the contrary, if destination address does not exist, then encryption will not be performed.
4. The web server will send the e-mail to the receiver's mail server.

Secure E-Mail Software

Requirement Analysis

Software requirements analysis is performed using Unified Modeling Language (UML).[15] The process involves determining actors and use cases. Actor is someone or something that interacts with the system being developed. An actor can be a human being, hardware or another system. Use cases are services or functions provided by the system to its users. A use case describes the behavior of the system, including the interaction between the actor and the system.

The actors inside the secure e-mail software are:

- The web user, the actor that uses the software to compose, read, send, and receive e-mail;

- The administrator, the actor who is in charge with administering the server;
- The mail server, the actor that represents the mail server and performs sending and receiving e-mail.

The functions that are required from the software are listed as follows:

- Login, to authenticate the user who uses the system;
- System maintenance, to list accounts, to create, and register request to create an account;
- E-mail encryption/decryption, to perform e-mail encryption/decryption process;
- Registering Account, to register new user;
- Composing mail, to create an e-mail message;
- Reading mail, to read an e-mail massage;
- Sending mail, to send an e-mail message to the mail server using SMTP protocol;
- Retrieving mail, to retrieve e-mail from the mail server using POP3 protocol;
- Managing the address book, to add or delete e-mail addresses from the address book;
- Managing an account, to view or change a user account.

Based on the above-described functions, the use case diagram shown in Figure 3 can be created.

Testing

Testing[16] is conducted to see the effect of encryption on using e-mail. The size of the e-mail and the time needed for encryption become factors to be observed. Based on the testing process, encryption will increase the average e-mail size to 161.96%, with average time for encryption about 3.68 seconds.

Conclusion

The research described in this article concludes that:

- The Risk Management approach is useful to determine the security requirements that will be implemented in an organization. By performing risk analysis, an organization can identify risks, determine why risks occur, decide priority based on the risk, and create a security strategy to reduce the risks.

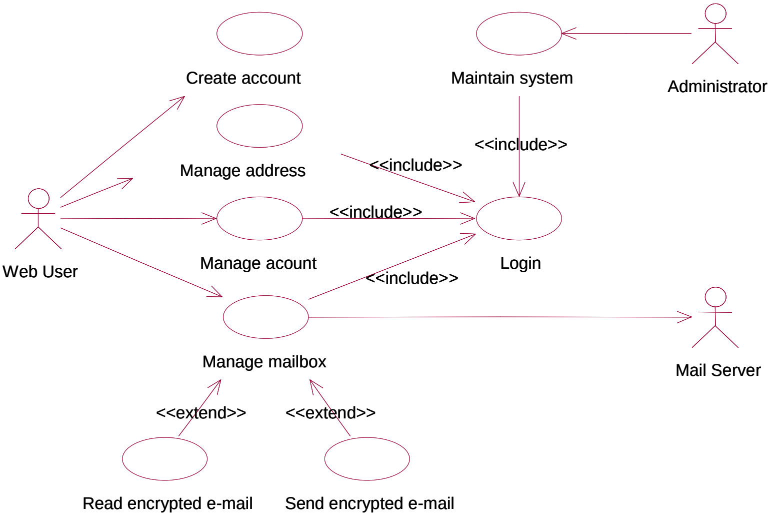

Figure 3: Use Case Diagram.

- The proposed security strategy can reduce the risks of eavesdropping, identity theft, message modification, false message, and denial of service.
- The encryption process will increase the average e-mail size to about 161.96%, with an average time for encryption about 3.68 seconds.

In accordance to the developed software, further progress needs to be done in order to reduce all possible risks.

Notes:

[1] *National Policy and Strategy for e-Government*, Indonesian's President Instruction No. 3 (2003), <http://www.ri.go.id/produk_uu/produk2003/ip2003/ip3'03.htm> (10 May 2004).

[2] Anthony Ralston, Edwin D. Reilly, and David Hemmendinger, eds., *Encyclopedia of Computer Science*, 4th edition (John Wiley & Sons, June 2000).

[3] Alferd J. Menezes, Paul C. van Oorschot, and Scott A. Vanstone, *Handbook of Applied Cryptography* (CRC Press, October 1996).

[4] Menezes, van Oorschot, and Vanstone, *Handbook of Applied Cryptography*.

[5] Bruce Schneier, *Applied Cryptography: Protocols, Algorithms, and Source Code in C* (John Wiley & Sons, 1995).

[6] Schneier, *Applied Cryptography: Protocols, Algorithms, and Source Code in C*.

[7] William Stallings, *Network and Internetwork Security: Principles and Practice* (Prentice Hall, 1995).

[8] Chistopher Alberts and Audrey Dorofee, *Managing Information Security Risks: The OCTAVE Approach* (Addison Wesley Professional, 2002).

[9] Alberts and Dorofee, *Managing Information Security Risks*.

[10] Alberts and Dorofee, *Managing Information Security Risks*.

[11] Gary Stoneburner, Alice Goguen, and Alexis Feringa, *Risk Management Guide for Information Technology Systems* (National Institute of Standards and Technology, July 2002), <http://csrc.nist.gov/publications/nistpubs/800-30/sp800-30.pdf> (25 November 2004).

[12] Stoneburner, Goguen, and Feringa, *Risk Management Guide for Information Technology Systems*.

[13] Setiyo Cahyono, *Development of Secure e-Mail Prototype*, Master Thesis in Information System (Informatics Department, Institute of Technology Bandung-Indonesia, 2004).

[14] Stephen A. Thomas, *SSL and TSL Essentials: Securing the Web* (John Wiley & Sons, 2000).

[15] Wendy Boggs and Michael Boggs, *Mastering UML with Rational Rose 2002* (Sybex Inc., January 2002).

[16] Roger S. Pressman, *Software Engineering: A Practical Approach* (McGraw Hill International, 1997).

KRIDANTO SURENDRO is Head of the Information Systems Laboratory Department of Informatics Engineering Institute of Technology, Bandung, Indonesia. He graduated from the Institute of Technology, Bandung, in 1987, and received a Ph.D. degree in Computer Science from Keio University, Japan, in 1999. He is one of the nationally recognised experts in Information Systems in Indonesia. *Address for correspondence*: Information System Laboratory, Department of Informatics Engineering, Institute of Technology Bandung, Jl. Ganesa 10, Bandung 40132, Indonesia; *Phone:* 62-22-250 8135; *E-mail:* endro@itb.ac.id.

SETIYO CAHYONO is a M.Sc. student at the Information Systems Department of Informatics Engineering Institute of Technology Bandung. He is expected to graduate his Master programme in 2004. *Address for correspondence*: Information System Laboratory, Department of Informatics Engineering, Institute of Technology Bandung, Jl. Ganesa 10, Bandung 40132, Indonesia.