

Monitoring employees' emails without violating their privacy right

Giannakis Antoniou¹, Udaya Parampalli², Lynn Batten³

^{1,2}The University of Melbourne, ³Deakin University
{gant, udaya}@csse.unimelb.edu.au, lmbatten@deakin.edu.au

Abstract

The capability of an employee to violate the policy of an organization is a concern for an employer. Monitoring is a measure taken by an employer to discourage an employee from acting inappropriately. However, current monitoring techniques tend to raise privacy issues because they violate the privacy rights of employees. Applying a monitoring technique without violating the privacy of employees is the aim of this paper. We propose a design and a protocol which give an employer the opportunity to monitor employee email in order to detect company policy violations. This can be achieved without violating the privacy of honest employees, while at the same time revealing evidence about the illegal actions of dishonest employees.

1. Introduction

Information systems (IS) are widely used in organizations and usually accessible from non-trusted networks, e.g. the Internet. In such cases an IS has a large number of potential attackers threats coming from inside the organisation (internal threats) which should not be underestimated [1]. There are studies [2] showing the high frequency and impact of insider attacks on an organization's IS. The impact and the frequency of policy violations by employees have driven organizations to find measures to detect and prevent them. In an attempt to discourage insider personnel from acting illegally, a monitoring system can be effective [2]. However, monitoring systems which violate the privacy rights of personnel raise privacy concerns, and are not acceptable in some countries such as the European Union [3]. On the one side, an employer wants to control the actions of employees in order to detect actions which violate the policy. On the other side, employees want to enjoy

their rights to privacy and to believe they are trusted by the employer.

In this paper we propose an email monitoring technique which allows internal users/personnel to keep their emails confidential as long as they do not violate the policy of their organisation. The structure of the paper is as follows: in Section 2 we introduce our email monitoring technique, in Section 3 we analyse it and we conclude the paper in Section 4.

2. Methodology

The aim of the proposed solution is to prevent and detect illegal emails (emails which violate the agreed policy of the organisation) from being exchanged without violating the privacy of honest personnel. An agent, which is located at the side of the user (in this case, the employee), and the Server Agent (SA), which is located at the side of the employer, co-operate before the employee sends or receives any email message. The email server is located at the side of the employer. For simplicity, we assume that the SA plays the role of the email server as well.

The agent and the SA are provided by a trusted third entity, such as a government authority, which is the owner of the software applications. This entity is responsible for checking regularly whether the agent has been modified. However, neither the employee nor the employer needs to rely on this authority because the actions of the agent and the SA are accountable. The agent can calculate and store information such as secret keys. It does not have access to the resources of the user's computer, apart from information exchanged by the user. Moreover, the agent does not have access to the communication means. Only the user is responsible for sending and receiving messages. Whenever the agent wants to send a message to the

SA, the agent must give the message to the user and the user will forward it.

The notation used in this paper is as follows:

$\{msg\}_{K_{SaKey}}$: The msg is encrypted (symmetric) by using the aKey.

$\{msg\}_{K_{pubKey}}$: The msg is encrypted (asymmetric) by using the pubKey.

$\{cipher\}_{DS_{aKey}}$: The cipher is decrypted (symmetric) by using the aKey

$\{cipher\}_{D_{privKey}}$: The cipher is decrypted (asymmetric) by using the privKey

$S_{privKey}(msg)$: The msg is signed using the privKey.

$IsEncrypted(msg)$: The function returns true if the msg is encrypted. Otherwise it returns false.

$IsLegal(msg, Version)$: The function returns true if the msg does not violate the rules of the specific Version (each version represents a different set of rules). Otherwise, it returns false.

2.1 Steps in the proposed protocol

We divide the protocol into two parts. The first part takes place when the user is sending an email; the second part takes place when the user is receiving an email. Therefore, an employer will be able to detect violation of the policy for incoming and outgoing emails. An agent has its own pair of keys (AGENTpublic-key/AGENTprivate-key). A user has no access to the private key of the agent. A user has two pairs of keys. The first key-pair (USER-AGENTpublic-key/USER-AGENTprivate-key) is given to the agent only to encrypt and decrypt messages on behalf of the user. The second key-pair (USERpublic-key/USER-private-key) is used only for signing and verifying messages. There are two reasons for letting the user have two pairs of keys:

- Allows the user (and a remote communication entity) to use an asymmetric cryptosystem, while the agent is able to decrypt and examine incoming emails.
- Prevents the agent from signing a message on behalf of the user. The agent knows only the USER-AGENTprivate-key; therefore, the agent can only encrypt and decrypt messages. The agent cannot

sign on behalf of the user without knowing the USERprivate-key.

2.1.1. Part 1 - Sending a message

Figure 1 illustrates the participating entities for each step during the sending procedure, and follows the actual protocol. During the first 3 steps, the agent and the SA share a number of session secret keys.

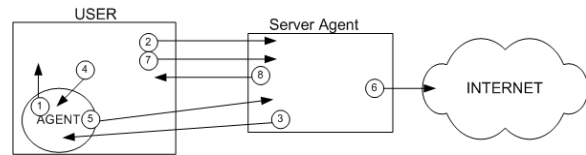


Figure 1 - The process while sending an email

- 1) The agent asks the user to request keys from the SA
Agent → User: "request keys"

The agent asks the user to request session keys when the agent has few or none unused session keys available. It is up to the user to contact the SA in order to request session keys. Without session keys, the agent cannot encrypt and send messages to the SA. The agent cannot even decrypt and forward an incoming message to the user.

- 2) The user requests a number (NumKeys) of keys from the SA. Each key (session key) will be used by the agent to encrypt the messages (step 5) during a communication session between the agent and the SA.

User → SA: $S_{USERprivate-key}$ ("request", NumKeys)

- 3) The SA sends the keys to the agent. Although the user has access to the exchanged information, he cannot have access to the key, because it is encrypted with the public key (AGENTpublic-key) of the agent.

SA → Agent: $S_{SAprivate-key}(ID, \{key\}K_{AGENTpublic-key})$

- 4) The user signs and sends the desired email as well as any cryptographic parameter to the agent. The DataMsg is the actual email. The CryptoPar could be anything (apart from the USERprivate-key). For example, the CryptoPar could contain the SERVERpublic-key. The SERVERpublic-key is the public key of the destination entity which the user wants to communicate with. Instead of the server's public key, it could be a secret key. In case the user wants to offer non-repudiation of the exchanged message, he can include in the CryptoPar the signed value of the DataMsg or the signed value of the CipherDataMsg.

User → Agent: $S_{USERprivate-key}(DataMsg, CryptoPar)$

The agent determines whether the DataMsg is encrypted. If the DataMsg is encrypted, then the agent will stop processing the user's request. Otherwise, the agent examines the validity of the DataMsg and in case the DataMsg is legal, it uses the parameters given by the user for encryption/signing. This encryption prevents the SA having access to the DataMsg. If the DataMsg has been characterized as illegal, then the agent will not encrypt the message based on the requirements (CryptoPar) of the user: Suppose the parameter includes the public key of the SERVER because the user wants to offer message confidentiality. The following algorithm shows the actions of the agent.

```

IF NOT isEncrypted (DataMsg) AND isLegal
(DataMsg, Version) THEN
  CipherDataMsg= {DataMsg}KSERVERpublic-key
  Msg= CipherDataMsg
ELSE IF NOT isLegal (DataMsg, Version) THEN
  Msg=SUSERprivate-key(DataMsg, CryptoPar)
ELSE Stop processing the user's request
END IF

```

5) Before sending the Msg to the SA, it encrypts the Msg with a key. The key has been given by the SA on step 3. However, the user has no access to that key.
 Agent →SA: S_{AGENTprivate-key}({Msg, Flag}K_{Skey}, ID, Version)

Based on the given ID, the SA knows which key the agent has used to encrypt the message. The agent is required to use the ID as described in step 3. Otherwise, the agent could have a hidden communication with the SA and act maliciously. The user is able to check the order of the IDs. This communication is the most critical from the privacy violation point of view. The agent sends an encrypted message to the SA. The agent could act maliciously and reveal private information of the user, such as a credit card number found in the user's computer. However, in a later step (step 8) we require that the SA must provide enough evidence about the content of that message. Therefore, the sent message of the agent is accountable, and any malicious attempt to violate the privacy of the user will be detected.

The value of the Flag (found in step 5) represents the identities (RuleID) of the rules the message violates. If there is no violation of any rule then the value of the Flag is zero. In case the Flag is zero, the SA has no access to that Msg because it is encrypted. However, in case the Flag is not zero, it means that the

Msg has been characterized (by the agent) as illegal and the Msg is not encrypted.

6) The SA decrypts the Data and forwards the Msg to the destination, only in case the Msg is legal.

SA →SERVER: Msg

Although the SA is able to identify whether a Msg is malicious or not based on the Flag, the SA is not able to have access to the content of the Msg, in case the Msg is encrypted. The SA will be able to have access to the content of the Msg only if the Msg is malicious.

7) The user asks the SA to provide evidence showing to the user that the exchanged messages did not violate the privacy of the user

User→SA: S_{USERprivate-key}(ID)

8) The SA sends the evidence

SA →User: {S_{Sprivate-key}(key, ID, Version)}K_{USERpublic-key}

The user verifies whether the agent has sent (in step 5) private information to the SA. Also, the Flag must be zero if the message was considered legal. Otherwise, the Flag contains the IDs of the rules that the message violates. If the key given in step 8 can decrypt the [{Msg, Flag}K_{Skey}] from step 5 and the Msg is indeed the expected one, then the key (from step 8) is the one used. Otherwise, it is not the valid key.

2.1.2. Part 2 - Receiving a message

Figure 2 illustrates the participating entities for each step during the receiving procedure, and follows the actual protocol.

9) The SA receives a message from an Internet user. Based on the DestEmailAddress, the SA determines the destination of the email.

ReceivedMsg=DestEmailAddress+

{ReceivedDataMsg}K_{USER-AGENTpublic-key}

Remote User →SA: ReceivedMsg

10) The SA encrypts, signs and forwards the received message to the appropriate agent through the related employee.

SA →Agent: S_{Sprivate-key}(ID, {ReceivedMsg}K_{Skey})

11) The agent uses the USER-AGENTpublic-key encrypting and accessing the ReceivedDataMsg. The agent decrypts the ReceivedDataMsg by using the USER-AGENTpublic-key and checks it if it is legal.

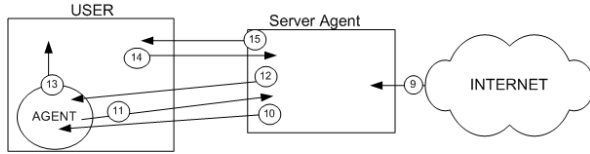


Figure 2 - The process while receiving an email

The Counter is a number which is increasing by one each time

```

IF the ReceivedMsg is not illegal THEN
  Comments= Counter
ELSE
  Comments=Violated RuleID + SecurityParameters
END IF

```

If the agent detects an illegal message, the SA is informed about the violated rule as well as the necessary security parameters to help SA decrypt the encrypted ReceivedMsg.

Agent \rightarrow SA: $S_{AGENTprivate-key}(\{Comments\}K_{Skey}, ID, Version)$

12)The SA confirms that he received the comments. This is a necessary step because a malicious user could prevent the message from step 11 reaching the SA.

SA \rightarrow Agent: $S_{SAprivate-key}(\{Comments\}K_{Skey}, ID, Version)$

13)If the ReceivedMsg is not illegal, the agent will give the ReceivedMsg to the user. Otherwise, the agent will not give it to the user.

Agent \rightarrow User: ReceivedMsg

14)The user asks the SA to provide evidence showing to the user that the exchanged messages between the agent and the SA didn't violate the privacy of the user

User \rightarrow SA: $S_{USERprivate-key}(ID)$

15)The SA sends the evidence, where the user verifies whether the agent has violated (in step 5) the privacy of the user or not.

SA \rightarrow User: $\{S_{SAprivate-key}(key, ID, Version)\}K_{USERpublic-key}$

3. Analysis

Using steps 5 and 11, the user is able to determine if there has been a privacy violation on the part of the SA. If there is not clear proof that there was no privacy violation, the user can claim and prove to any third entity that the SA violated his privacy. We analyze both the send and receive procedures to indicate where the user might proof of, or the absence of proof of, a privacy violation.

The user may have suspicions about a privacy violation during the communication by examining mainly the results of steps 3, 5 and 11. However the user can use the received key from the SA (in steps 8 or 15) to find out whether there was a privacy violation or not. We divide the analysis into two parts. In the first part, we examine the send procedure and in the second part we examine the receive procedure.

3.1 Send Procedure

The following algorithm can be used by the user in order to verify whether there was a privacy violation or not during the send procedure. In order to explain the algorithm more clearly, we provide in brackets the number of the step in which we find the related information. E.g. key <5> refers to the key found in step 5.

Let $ExpectedMsg = \{DataMsg<4>\}K_{SERVERpublic-key}$

IF key<8> can decrypt $\{Msg, Flag\}K_{Skey}<5>$ AND $Msg = ExpectedMsg$ THEN

The key<8> is the correct one

IF the flag=0 THEN

The Msg has been characterized as legal and the Msg has not been revealed to the SA (if the Msg was encrypted)

ELSE

The Msg has been characterized as illegal and the SA had access to the Msg. The Flag has the violated RuleID (one or more)

END IF

ELSE

The key<8> is incorrect or the agent has encrypted the Msg in inappropriate way. Therefore, the user can take further actions by accusing the SA.

END IF

The user can verify the encrypted message found in step 3 by doing the following: The user encrypts the key<8> with the public key of the agent

Let $X = \{key<8>\}K_{AGENTpublic-key}$

If X is equal to $\{key<3>\}K_{AGENTpublic-key}$ then key<8> is equal to key<3>. Although the user does not know the private key of the agent, he can verify the encrypted content of the message from step 3. The proposed technique offers confidentiality to the email however, the destination email address is known by the employer.

3.2 Receive Procedure

A user who is receiving inappropriate email is not responsible for the exact message. However, an incoming inappropriate email will not reach the destination employee. The following algorithm can be used by the user in order to verify whether there was a privacy violation or not during the send procedure.

```
IF  $\{\{ReceivedMsg\}KS_{key<10>}\}DS_{key<15>}$  =  
ReceivedMsg<13> THEN  
  key<10> = key<15> and ReceivedMsg<10> =  
  ReceivedMsg<13>  
  X =  $\{\{Comments\}KS_{key<11>}\}DS_{key<15>}$   
  IF (X violates the privacy of the user AND  
  ReceivedMsg<13> does not violate any of the  
  agreed rules) OR  
   $\{Comments\}KS_{key<11>}$  !=  
   $\{Comments\}KS_{key<12>}$  THEN  
    there is an unreasonable privacy violation  
  ELSE No privacy violation exist  
ELSE there is an unreasonable privacy violation  
END IF
```

3.3 Accusing the Agent/SA

Once the user realizes that the agent acted maliciously, the user can accuse the SA. Even though the malicious actions were executed by the agent application, the owner of the agent is responsible for them. For this reason, a tamperproof technique offered by the agent can be helpful. In case the agent has sent an unexpected message, then the user can provide the related information (steps 3, 4, 5 and 8 for sending a message) to a third entity, such as the police, and prove that the agent has acted maliciously. If the user claims that the Msg<5> is not the expected (ExpectedMsg) then the police will ask the SA (or the agent) to provide evidence on how the agent has constructed the MSG<5>. The police may actually ask for information from step 4, where the user signs the message he wanted to send. Based on the message of step 4 and the key<5> (where key<5> is equal to key<8>), the Police can construct the expected message (ExpectedMsg). If the SA has no evidence, then the agent has acted maliciously. If the user claims to the police that he requested the key from the SA, and the SA did not give the correct key (based on the related ID), then the government authority can determine whether the user has this right or not, based on the information from steps 3 and 8. The police encrypts the key<8> with the public key of the agent (AGENTpublic-key).

$X = \{key<8>\}K_{AGENTpublic-key}$

IF X is equal to $\{key\}K_{AGENTpublic-key}<3>$, where
 $ID<3>=ID<8>$, THEN

The key<8> is equal to key<3>; therefore the
SA gave the correct key.

ELSE The key<8> is not equal to key<3>; therefore
the SA gave incorrect key.

END IF

4. Conclusion

The desire of employers to detect the violation of company policy by employees along with privacy concerns of employees due to the measures taken by employers has raised an interesting problem. In this paper we introduced a technique which allows an employer to detect a policy violation related to a dishonest employee's exchange of emails while at the same time, an honest employee can keep his/her exchanged emails private. Moreover, an employer may collect evidence about the malicious actions of dishonest employees while an employee can also accuse his/her employer in case the employer tries to violate the privacy of an honest employee.

In further work, submitted elsewhere, we present several scenarios allowing us to extend these ideas, including a dynamic version of the solution.

5. References

- [1] R. Willison, "Understanding the perpetration of employee computer crime in the organisational context," *Information and Organization*, vol. 16, pp. 304 - 324, 2006.
- [2] S. Fleming, "Implicit Trust Can Lead to Data Loss," *Information Systems Security*, vol. 16, pp. 109 -113, 2007.
- [3] L. Mitrou and M. Karyda, "Employees' privacy vs. employers' security: Can they be balanced?," *Telematics and Informatics*, vol. 23, pp. 164-178, 2006.