7-2005

Deposit-case attack against secure roaming

Guomin YANG
Singapore Management University, gmyang@smu.edu.sg

Duncan S. WONG
City University of Hong Kong

Xiaotie DENG
City University of Hong Kong

Citation

YANG, Guomin; WONG, Duncan S.; and DENG, Xiaotie. Deposit-case attack against secure roaming. (2005). *Information Security and Privacy: 10th Australasian Conference, ACISP 2005, Brisbane, Australia, July 4-6: Proceedings*. 3574, 417-428.
Available at: https://ink.library.smu.edu.sg/sis_research/7440

Deposit-Case Attack Against Secure Roaming

Guomin Yang, Duncan S. Wong*, and Xiaotie Deng

Department of Computer Science
City University of Hong Kong
Hong Kong, China
{csyanggm,duncan,deng}@cs.cityu.edu.hk

Abstract. A secure roaming protocol involves three parties: a roaming user, a visiting foreign server and the user's home server. The protocol allows the user and the foreign server to establish a session key and carry out mutual authentication with the help of the home server. In the mutual authentication, user authentication is generally done in two steps. First, the user claims that a particular server is his home server. Second, that particular server is called in by the foreign server for providing a 'credential' which testifies the user's claim. We present a new attacking technique which allows a malicious server to modify the user's claim in the first step without being detected and provide a fake credential to the foreign server in the second step in such a way that the foreign server believes that the malicious server is the user's home server. We give some examples to explain why it is undesirable in practice if a roaming protocol is vulnerable to this attack. We also show that there are three roaming protocols proposed previously which are vulnerable to this attack.

Keywords: Protocol Security Analysis, Authenticated Key Exchange, Roaming

1 Introduction

With the rapid development of mobile technologies, user mobility is becoming an important network feature nowadays. People can travel around with their mobile devices without being limited by the geographical coverage of their home networks. They can access different foreign networks, identify themselves as subscribers of their home networks and get access to the foreign networks after passing some authentication procedures. This scenario is called *roaming*.

A typical roaming scenario involves three parties: a *roaming user*, A, a visiting *foreign server*, V, and the user's *home server*, H. The roaming user A subscribed to the home server H is now in a network operated by the foreign server V. A communicates directly with V but does not have direct link with H. On the other hand, V has direct link with H. Before allowing A to connect to V, the

* The work was supported by a grant from the Research Grants Council of the Hong Kong Special Administrative Region, China (Project No. 9040904 (RGC Ref. No. CityU 1161/04E)).

C. Boyd and J.M. González Nieto (Eds.): ACISP 2005, LNCS 3574, pp. 417–428, 2005.

foreign server V first finds out the identity of A's home server and obtains a valid 'credential' from the home server which testifies that A is a legitimate subscriber of the home server. In other words, V does not authenticate A directly. Instead, H authenticates A via V and then provides a credential to V for testifying the authentic subscription of A.

Roaming services have been widely deployed in cellular networks such as [8, 11] and 3GPP[1]. Besides mobile communications, there are many other systems and applications that can be considered as roaming in protocol perspective. Some of them are actually roaming on wired networks. In [1], Ateniese, et al. gave two examples. One is the inter-bank ATM networks and the other is the credit card payment systems. In an inter-bank ATM network, a customer (a roaming user) comes to an ATM machine[2], which is not operated by the customer's bank, and accesses his bank account. Financial transactions such as withdrawing and depositing money are provided by the ATM machine after the machine has obtained enough assurance on the customer's good standing with respect to his ATM card. Similar roaming environment exists in a credit card payment system by considering the merchant's bank as the foreign server and the credit card issuing bank as the home server of the credit card holder. There are some emerging technologies which can also be modeled as roaming. For example, hopping across meshed WLANs (Wireless Local Area Networks) administered by different individuals, joining and leaving various wireless ad hoc networks operated by different foreign operators, etc.

On the security of roaming, almost all secure roaming protocols support *Subscription Validation* and *Key Establishment*.

Subscription Validation is satisfied if the following conditions are satisfied.

1. The foreign server is sure about the home server of the user.
2. The foreign server gets some 'credential' from the home server of the user which testifies that the user is a legitimate subscriber of the home server.

Key Establishment allows the foreign server and the roaming user to share a session key which is used to secure the communication channel between them. There are some other security requirements for some specific roaming protocols. For example, the latest cellular system 3GPP requires *Server Authentication* which allows the roaming user to authenticate the visiting foreign server. Some other roaming protocols [1, 10, 12, 5] also consider *User Anonymity and Untraceability* as required security objectives. These additional security requirements enable users to roam anonymously without being located or tracked.

Among these security requirements, Subscription Validation is intuitively related to the financial interests of the foreign server and the home server of the user. By getting a credential from the home server of the user, a foreign server is able to request the user's home server for service charge as the credential becomes a proof for payment request. In order to protect the financial interests of both the foreign server and the home server, the credential is required to

[1] http://www.3gpp.org

[2] For example, an ATM terminal with Visa/PLUS or Mastercard/Cirrus sign on.

be secure against forgery and it should also be one-time so that the credential cannot be replayed.

In this paper, we show that Subscription Validation is also related to the financial interest of the user. We present a new attacking technique which incurs the following two results simultaneously.

1. The attack allows a malicious server to persuade the visiting foreign server of a roaming user that the malicious server is the user's home server without being noticed by the user nor the real home server.
2. The roaming user, however, believes that the foreign server has obtained the correct value about the identity of his home server.

We call this attack the **Deposit-case Attack** as such attack is profitable to the malicious server in the case when the user is accessing the foreign server to 'deposit' some information of value (such as electronic cash) to his home server.

The first impression one may have on the deposit-case attack is that it is similar to an Unknown Key Share Attack [3]. In some cases, they cause similar damage. However in some other cases, they are different.

An unknown key share attack applies to a key agreement protocol [2]. It makes one party A believe that a session key is shared with a party B when it is in fact shared with another party C. If party B is the adversary, then the unknown key share attack causes similar damage on a key agreement protocol to that of the Deposit-case Attack on a roaming protocol.

However, a roaming protocol is not simply a kind of key agreement protocols. A roaming protocol can also be an authentication protocol when the Key Establishment between the roaming user and the foreign server is not required. In this case, the Deposit-case Attack against an authentication-only roaming protocol will make the user A believe that the foreign server V has obtained the identity of A's home server (i.e. H) when it has in fact obtained the identity of another server which is malicious.

The Deposit-case Attack is not well captured in the security requirements of current roaming protocols. Apparently, the attack may not even be considered in many of such protocols as we have found three roaming protocols [10, 5, 7] that are vulnerable to this attack. In the following, we give details of the deposit-case attack and explain how this attack could bring very undesirable consequences in practice (Sec. 2). Then we show that there are three roaming protocols that can be compromised by the deposit-case attack in Sec. 3, 4 and 5, respectively. We conclude the paper in Sec. 6 by discussing a corrective approach against this attack.

2 Deposit-Case Attack

In most of the current roaming protocols, Subscription Validation is done in two steps.

1. The roaming user A claims that a particular server H is his home server.
2. That particular server, H, is then called in as a guarantor by the visiting foreign server V for giving a promise (as a one-time unforegable credential) that A is one of H's legitimate subscribers.

In the second step above, H generates a credential only after authenticating A. This effectively prevents the following attack.

Consider a malicious user B, who is not subscribed to any server, claims that a server, H, is his home server and manage to create a fake credential which results to have the foreign server V believe that H is B's home server. This attack directly conflicts with the interest of H if the Subscription Validation protocol is vulnerable to this attack. Most of the current roaming protocols have the two-step Validation Subscription mechanism described above implemented to thwart this attack.

We now consider a new attacking scenario which is called the Deposit-case Attack against roaming protocols. In this scenario, the user is honest while there is a malicious server[3], M. Suppose the user's home server is H. The malicious server M will make the foreign server V believe that the home server of the user is M without being detected by the user nor the real home server H of the user.

Notice that the two-step Subscription Validation mechanism described above may not be able to prevent the Deposit-case Attack because when the foreign server receives a valid credential from the malicious server, there is no guarantee that the user's claim in the first step is not modified. Suppose the malicious server M modifies the user's claim in the first step and produces a one-time unforgeable credential to the foreign server in the second step. This can be done by M as M is also a server in the system. Consequently, the foreign server believes that M is the user's home server. In this attack, the user believes that he has correctly informed the foreign server that his home server is H while the foreign server believes that the home server of the user is the malicious server M.

2.1 Practical Impacts of the Deposit-Case Attack

It is undesirable if a roaming protocol is vulnerable to the Deposit-case Attack. This attack is profitable to the malicious server in the case when the user is accessing the foreign server to 'deposit' some information of value (such as electronic cash) to his home server. Since the foreign server believes that the user is a subscriber of the malicious server, credit for this deposit will go to the malicious server.

Consider the roaming environment of an inter-bank ATM system described in Sec. 1. Suppose there is a roaming user using an ATM machine operated by a foreign bank (i.e. a foreign server) to deposit money to his bank account located at the user's bank (i.e. the user's home server). If the ATM system is vulnerable

[3] The malicious server can also be viewed as a malicious 'insider' [6] of the underlying roaming system.

to the deposit-case attack, we can see that it would allow the foreign bank to transfer money to a malicious bank instead of the user's bank account.

To some extent, the Deposit-case Attack causes similar damage on a roaming protocol to that of an unknown key share attack on a key agreement protocol [3]. But they are also different as explained in Sec. 1. In the following, we will see that protocols of [10, 5, 7] cannot defend themselves against the Deposit-case Attack.

3 An Anonymous Roaming Protocol

In [10], Samfat et al. proposed a suite of protocols for secure roaming. Besides Server Authentication, Subscription Validation and Key Establishment, their protocols also support certain degrees of User Anonymity and Untraceability. All of their protocols are derived from one basic protocol. In the following, we first review their basic protocol and show that it is vulnerable to the deposit-case attack. The attacking technique can be applied directly to all of their other protocols.

Let E_K be the encryption function under the symmetric key K. The symmetric key encryption function is assumed to be a block cipher (e.g. AES [9]). We use PKE_A to denote the public key encryption function of party A and Sig_A to denote the signature generation function of A. The $\oplus$ symbol indicates a bitwise exclusive-OR operation and the $||$ symbol represents the binary string concatenation.

3.1 The Basic Protocol of Samfat et al.

There are two functions used as building blocks in the protocol: $Token_K$ and $TICK_K$. $Token_K$ is computed by applying a block cipher E_K over three inputs: m_1, m_2 and m_3.

$$Token_K(m_1, m_2, m_3) = E_K(m_1 \ \oplus \ E_K(m_2 \ \oplus \ E_K(m_3))).$$

$TICK_K$ is called a ticket which is used by an initiator A for sending a session key σ to a responder B. The key is also intended to be shared with a third party C. This is denoted by

$$TICK_K(A, B, C, \sigma) = Token_K(N_1 \oplus C, \ N_2, \ N_1 \oplus A) \oplus \sigma$$

where N_1 and N_2 are nonces that are randomly generated.

Let A be a roaming user, V be a foreign server and H be the home server of the roaming user. The Basic Protocol of Samfat et al. consists of four message flows among these three parties. The fourth message flow is optional. In the following, we first describe the protocol with the first three message flows only. We will consider the fourth message later.

$$
\begin{aligned}
&A \rightarrow V : H,\ alias = PKE_H(N_1 \ ||\ N_1 \oplus A),\\
&\qquad\quad AUTH_1 = \langle\ N_2,\ T,\ Token_{K_{av}}(alias, T, N_2)\ \rangle\\
&V \rightarrow H : alias,\ PKE_H(N_3 \ ||\ N_3 \oplus V),\\
&\qquad\quad AUTH_2 = \langle\ N_4,\ AUTH_1,\ Token_{K_{vh}}(V, AUTH_1, N_4)\ \rangle\\
&H \rightarrow V : PKE_V(N_3),\ TICK_{K_{vh}}(H, V, alias, K_{av})
\end{aligned}
$$

In the protocol, N_1, N_2, N_3, N_4 are nonces. T is a timestamp generated by A. $K_{av} = \mathcal{H}(A||V||K_{ah})$ where K_{ah} is a long-term key shared by A and H, and $\mathcal{H}$ is a one-way hash function. K_{vh} is a long-term key shared by V and H. By $\langle m_1, m_2 \rangle$, we mean some appropriate encoding of two messages m_1 and m_2.

3.2 Deposit-Case Attack

The attack described below follows directly the attacking technique delineated in Sec. 2. In the attack, we consider that there exists a malicious server M. The malicious server M first modifies the user's claim by replacing H with M in the first message flow from A to V. Then when V asks M for a credential, which corresponds to the second message flow, M generates and sends back the third message flow as a credential. As a result, V believes that M is the user's home server without being known by the user A. Below are the details of the attack.

The malicious server M intercepts the message from A to V and launches the following attack.

$$
\begin{aligned}
&A \rightarrow M : H,\ alias = PKE_H(N_1 \ ||\ N_1 \oplus A),\\
&\qquad\quad AUTH_1 = \langle\ N_2,\ T,\ Token_{K_{av}}(alias, T, N_2)\ \rangle\\
&M \rightarrow V : M,\ alias,\ AUTH_1' = \langle\ N_2',\ T',\ Token_{K'}(alias, T', N_2')\ \rangle\\
&V \rightarrow M : alias,\ PKE_M(N_3 \ ||\ N_3 \oplus V),\\
&\qquad\quad AUTH_2' = \langle\ N_4,\ AUTH_1',\ Token_{K_{vm}}(V, AUTH_1', N_4)\ \rangle\\
&M \rightarrow V : PKE_V(N_3),\ TICK_{K_{vm}}(M, V, alias, K')
\end{aligned}
$$

N_2' is a nonce, T' is a timestamp and K' is a random symmetric key generated by M. K_{vm} is a long-term key shared by V and M.

In the attack, A believes that he has informed V that his home server is H while V believes that the home server of A is M.

We now consider the optional fourth message flow. The purpose of this message flow is to allow V to send its public key to A so that the public key can be used for authentication in the future. Let the public key of V be P_V. The fourth message is denoted by

$$TICK_{K_{av}}(V, alias, V, P_V).$$

In the deposit-case attack, the fourth message will become

$$TICK_{K'}(V, alias, V, P_V).$$

Due to the lack of message authentication, we can see that A will still accept, but just get the wrong P_V. Therefore, the deposit-case attack still works.

4 Another Anonymous Roaming Protocol

In [5], Go and Kim proposed a different roaming protocol which targets to achieve the similar set of security goals to that of Samfat et al. reviewed in Sec. 3.

Let (G, g, q) be the domain parameters where $G = \langle g \rangle$ and the order of G is a large prime q. Assume the discrete logarithm problem in G is hard. Let A, V, H denote a roaming user, a foreign server and the home server of the user, respectively. We use the same set of notations as in Sec. 3. Let $\mathcal{H}_1$ and $\mathcal{H}_2$ be some cryptographically strong hash functions. By $x \in_R X$, we mean that an element x is randomly chosen from the set X. Let $(\hat{S}_H, P_H) \in \mathbb{Z}_q \times G$ be H's private key/public key pair such that $P_H = g^{\hat{S}_H}$. Let $(\hat{S}_V, P_V) \in \mathbb{Z}_q \times G$ be V's private key/public key pair such that $P_V = g^{\hat{S}_V}$. Let T_1, T_2 and T_3 be timestamps. Assume the public keys of all parties are publicly known. The Go-Kim protocol is shown as follows.

$$
\begin{array}{ll}
A & : r_a \in_R \mathbb{Z}_q,\ K_{ah} = P_H^{r_a},\ alias = E_{K_{ah}}(\mathcal{H}_1(A) \oplus g^{r_a}) \\
A \rightarrow V & : H,\ alias,\ g^{r_a} \\
V & : r_v \in_R \mathbb{Z}_q \\
V \rightarrow H & : alias,\ g^{r_v},\ g^{r_a},\ Sig_V(g^{r_v}, g^{r_a}, alias, V),\ T_1 \\
H & : r_h \in_R \mathbb{Z}_q,\ K_{hv} = \mathcal{H}_2(g^{r_v r_h}, P_V^{r_h}) \\
H \rightarrow V & : g^{r_h},\ E_{K_{hv}}(Sig_H(g^{r_h}, g^{r_v}, \mathcal{H}_1(A) \oplus g^{r_a}, H), \mathcal{H}_1(A) \oplus g^{r_a}),\ T_2 \\
V & : alias' = \mathcal{H}_1(g^{r_v r_a}, \mathcal{H}_1(A)),\ K_{av} = \mathcal{H}_2(g^{r_v r_a}, g^{\hat{S}_V r_a}) \\
V \rightarrow A & : g^{r_v},\ E_{K_{av}}(\mathcal{H}_1(g^{r_v}, g^{r_a}, alias', V), T_2),\ T_3 \\
A \rightarrow V & : E_{K_{av}}(Sig_A(g^{r_a}, g^{r_v}, T_2, V), T_3)
\end{array}
$$

4.1 Deposit-Case Attack

Direct application of the attacking technique outlined in Sec. 2 would not work over here. This is because the malicious server M has to decrypt *alias* and obtain the real identity of A in order to deliver the correct value to V and let A accept when A receives a commitment of $alias'$ in the second last message flow. However, M does not know K_{ah} which is needed to decrypt *alias*.

Note that *alias* is used to hide the real identity of A so that the Go-Kim protocol can provide user anonymity and untraceability against eavesdroppers. Hence before launching the deposit-case attack, M should find out the real identity of A. Below are the details on how M can find out A's real identity[4] and launch the deposit-case attack. Let $P_M \in G$ be M's public key.

[4] Precisely, M finds out the value of $\mathcal{H}_1(A)$ in the attack. However, the commitment $\mathcal{H}_1(A)$ has already provided enough information for an adversary to trace and reveal the identity of the user.

$$
\begin{array}{ll}
A & : r_a \in_R \mathbb{Z}_q,\ K_{ah} = P_H^{r_a},\ alias = E_{K_{ah}}(\mathcal{H}_1(A) \oplus g^{r_a}) \\
A \rightarrow M & : H,\ alias,\ g^{r_a} \\
M & : r_1 \in_R \mathbb{Z}_q \\
M \rightarrow H & : alias,\ g^{r_1},\ g^{r_a},\ Sig_M(g^{r_1}, g^{r_a}, alias, M),\ T_0 \\
H & : r_h \in_R \mathbb{Z}_q,\ K_{hm} = \mathcal{H}_2(g^{r_1 r_h}, P_M^{r_h}) \\
H \rightarrow M & : g^{r_h},\ E_{K_{hm}}(Sig_H(g^{r_h}, g^{r_1}, \mathcal{H}_1(A) \oplus g^{r_a}, H), \mathcal{H}_1(A) \oplus g^{r_a}),\ T_2 \\
M \rightarrow V & : M,\ alias,\ g^{r_a} \\
V & : r_v \in_R \mathbb{Z}_q \\
V \rightarrow M & : alias,\ g^{r_v},\ g^{r_a},\ Sig_V(g^{r_v}, g^{r_a}, alias, V),\ T_1 \\
M & : r_2 \in_R \mathbb{Z}_q,\ K_{mv} = \mathcal{H}_2(g^{r_v r_2}, P_V^{r_2}) \\
M \rightarrow V & : g^{r_2},\ E_{K_{mv}}(Sig_M(g^{r_2}, g^{r_v}, \mathcal{H}_1(A) \oplus g^{r_a}, M), \mathcal{H}_1(A) \oplus g^{r_a}),\ T_2 \\
V & : alias' = \mathcal{H}_1(g^{r_v r_a}, \mathcal{H}_1(A)),\ K_{av} = \mathcal{H}_2(g^{r_v r_a}, g^{\hat{S}_V r_a}) \\
V \rightarrow A & : g^{r_v},\ E_{K_{av}}(\mathcal{H}_1(g^{r_v}, g^{r_a}, alias', V), T_2),\ T_3 \\
A \rightarrow V & : E_{K_{av}}(Sig_A(g^{r_a}, g^{r_v}, T_2, V), T_3)
\end{array}
$$

In this attack, the malicious server M first pretends to be a foreign server, contacts A's home server H, and claims that A is communicating with M. H then innocently sends A's real identity to M. After that, M launches the deposit-case attack by impersonating A and sending a modified message to V (illustrated as the first message from M to V in the diagram above). This message makes V believe that M is the home server of A while A believes that he has informed V that H is his home server. The attack is then carried out in the same way as described in Sec. 2.

Notice that A and V will still agree on the same key K_{av} when the attack completes. Hence the attack is carried out successfully and will not be discovered by any of the three honest parties.

5 A Self-encryption Based Roaming Protocol

We now describe the third roaming protocol which is found to be vulnerable under the deposit-case attack. The protocol was proposed by Hwang and Chang [7] in 2003. The parties involved in the roaming protocol include a roaming user A, a visiting foreign server V and the user's home server H. It is a symmetric key based protocol which requires a secure symmetric key encryption algorithm such as AES [9]. We use the same set of notations as previous sections. Let K_{ah} denote the long-term secret key shared by A and H. Let K_{vh} denote the long-term secret key shared by V and H. Let f be a secure hash function kept secretly by H. We review their protocol as follows.

1. A generates a random value r_0 and sends the message below to V.

$$A \rightarrow V \ : \ A,\ H,\ E_{K_{ah}}(K_{ah}||r_0)$$

2. V generates a random value r_1 and sends the following to H for verification.

$$V \rightarrow H \ : \ E_{K_{ah}}(K_{ah}||r_0),\ E_{K_{vh}}(A||r_1||t)$$

Here t denotes a timestamp.

3. H decrypts the received message. If t is fresh and K_{ah} is equal to $f(A)$, H sends the following message back to V.

$$H \rightarrow V \ : \ E_{K_{vh}}(r_1),\ C = E_{k_{ah}}(r_0||r_1||V)$$

   Otherwise, H rejects the connection.
4. V decrypts $E_{K_{vh}}(r_1)$. If the decrypted value equals r_1, V sets r_1 as the authentication key K_{auth} and passes C to A. Otherwise, V rejects the connection.
5. On receiving C from V, A checks whether the decrypted message contains r_0. If it is false, A terminates the connection. Otherwise, A also sets r_1 as the authentication key K_{auth} and sends the following to V for authentication.

$$A \rightarrow V \ : \ E_{K_{auth}}(r_1)$$

6. V accepts if the decryption of the incoming message is equal to r_1. Otherwise, V rejects the connection.

After establishing K_{auth} between A and V, the authentication process of all subsequent sessions between these two parties can be simplified in such a way that V does not have to ask H for verifying A. Instead, V can talk directly to A and carry out mutual authentication using K_{auth}. For simplicity and without contradicting any of the assumptions made in [7], we hereafter assume that the lengths of all the random numbers and the identities of A, H and V are equal to the block size of the underlying block cipher.

5.1 Deposit-Case Attack

In the following, we describe the deposit-case attack launched by a malicious server M against the Hwang-Chang roaming protocol reviewed above. The attack is slightly different from the one described in Sec. 4.1. This time, the malicious server M uses the user's home server H as an encryption oracle for generating some message which is expected by A from his home server H.

Let K_{mv} be the long-term secret key shared by M and V and K_{mh} be the long-term secret key shared by M and H. M intercepts the first message from A to V and launches the following attack.

$$\begin{aligned}
&A \rightarrow M : A,\ H,\ E_{K_{ah}}(K_{ah}||r_0)\\
&M \rightarrow V : A,\ M,\ E_{K_{ah}}(K_{ah}||r_0)\\
&V \rightarrow M : E_{K_{ah}}(K_{ah}||r_0),\ E_{K_{mv}}(A||r_1||t)\\
&M \rightarrow H : E_{K_{ah}}(K_{ah}||r_0),\ E_{K_{mh}}(A||r_1||t)\\
&H \rightarrow M : E_{K_{mh}}(r_1),\ C' = E_{K_{ah}}(r_0||r_1||M)\\
&M \rightarrow V : E_{K_{mv}}(r_1),\ C'\\
&V \rightarrow A \ : C'\\
&A \rightarrow V \ : E_{K_{auth}}(r_1)
\end{aligned}$$

Note that C' contains the identity of M. The crucial issue of arguing whether the attack works or not is to determine whether A will check the encrypted identity in C in Step 5 of Sec. 5. This is not mentioned in Hwang-Chang's protocol description [7]. We now consider the two possible cases.

1. If A does not check the identity (i.e. the last component) after decrypting C in Step 5 of Sec. 5, then the attack succeeds. The authors of [7] seem not checking it according to the description of their protocol.
2. If A checks the identity encrypted in C, then there are two sub-cases.
   (a) A finds out which foreign server he is talking to by checking the identity after decrypting C.
   (b) A intends to talk to V at the very beginning when A initiates the protocol execution.

Depending on whether A knows if he is talking to V or M at the very beginning of the protocol execution, in Case 2(a), A believes that he is talking to M after checking the identity in C' while V believes that he is talking to A. In addition, V believes that A's home server is M. Hence in Case 2(a), the deposit-case attack works.

For Case 2(b), A will reject the connection with failure if the deposit-case attack is launched. However, we will see that under some assumptions, the malicious server can still launch the deposit-case attack successfully by modifying the last two message flows of the attack described above slightly. Also, the assumptions made do not contradict any of the restrictions or assumptions made in [7].

In order to make the deposit-case attack work in Case 2(b), the malicious server M has to modify C' in such a way that it contains V as the last component of the corresponding plaintext. However, M does not know the value of K_{ah}.

This can be solved by looking into the implementation details of the underlying block cipher $E_{K_{ah}}$. For simplicity, let us assume that the operation mode [4] of the underlying block cipher is ECB (Electronic Codebook). The computation of C in Hwang-Chang protocol becomes

$$C = E_{K_{ah}}(r_0) \mid\mid E_{K_{ah}}(r_1) \mid\mid E_{K_{ah}}(V).$$

It is also the case for computing C' but with the last component being changed to $E_{K_{ah}}(M)$. Suppose there has been a successful protocol execution among A, V and H before M launches the Deposit-case Attack. Then M gets $E_{K_{ah}}(V)$ from the protocol execution through eavesdropping.

To launch the deposit-case attack, M intercepts the last message flow from V to A and replaces the last component of C' by $E_{K_{ah}}(V)$. We can see that A will accept and complete the protocol without early termination. Also notice that M knows the authentication key K_{auth} as M knows the value of r_1.

In this modification, we have made two assumptions.

1. There is at least one successful protocol execution among A, V and H, and M is able to eavesdrop that protocol execution.
2. The underlying block cipher $E_{K_{ah}}$ is operated in ECB mode.

None of these assumptions contradicts the restrictions or assumptions made in [7]. In addition, the second assumption can also be extended to other commonly used operation modes. It is obvious that if the operation mode is CBC (Cipher

Block Chaining), CFB (Cipher Feedback) or OFB (Output Feedback) [4], the malicious server M can still manage to make all the parties accept and complete the deposit-case attack. This is because M knows the last two components of the plaintext corresponding to C'. They are r_1 and M.

6 Concluding Remarks

We present a new attacking technique against secure roaming protocols. The attack allows a malicious server to make a user believe that the visiting foreign server has been informed about the true identity of the user's home server while the foreign server believes that the malicious server is the home server of the user. We explain that this attack is profitable to the malicious server if the protocol is used by the user to deliver some information of value (such as some electronic cash) to his home server via the foreign server. We also show that there are three roaming protocols proposed previously which are vulnerable to this attack.

There is no universal solution for these three roaming protocols so that they can thwart the deposit-case attack. However, there is a plausible approach which can be adopted when modifying these protocols. The approach is to have the roaming user check if the foreign server has obtained a valid credential from his real home server before accepting the connection. None of the three roaming protocols reviewed in this paper has done this checking. As more and more new roaming-like systems and applications are emerging, we believe that this new attack should be checked against if the corresponding systems and applications have related concerns discussed in this paper.

References

[1] G. Ateniese, A. Herzberg, H. Krawczyk, and G. Tsudik. On traveling incognito. In *Proc. of the IEEE Workshop on Mobile Systems and Applications*, December 1994.

[2] C. Boyd and A. Mathuria. *Protocols for Authentication and Key Establishment.* Springer-Verlag, 2003.

[3] Whitfield Diffie, Paul C. Van Oorschot, and Michael J. Wiener. Authentication and authenticated key exchanges. *Designs, Codes, and Cryptography*, 2(2):107–125, June 1992.

[4] M. Dworkin. *Recommendation for Block Cipher Modes of Operation.* NIST Special Publication 800-38A 2001 Edition, US Department of Commerce / NIST, December 2001.

[5] J. Go and K. Kim. Wireless authentication protocol preserving user anonymity. In *Proc. of the 2001 Symposium on Cryptography and Information Security (SCIS 2001)*, pages 159–164, January 2001.

[6] D. Gollmann. Insider fraud (position paper). In *Security Protocols Workshop*, pages 213–219. Springer, 1998. LNCS 1550.

[7] K. F. Hwang and C. C. Chang. A self-encryption mechanism for authentication of roaming and teleconference services. *IEEE Trans. on Wireless Communications*, 2(2):400–407, March 2003.

[8] Michel Mouly and Marie-Bernadette Pautet. *The GSM System for Mobile Communications*. Published by the authors, 1992.
[9] NIST FIPS PUB 197. *Announcing the ADVANCED ENCRYPTION STANDARD (AES)*, November 2001.
[10] D. Samfat, R. Molva, and N. Asokan. Untraceability in mobile networks. In *Proc. of MobiCom '95*, pages 26–36, 1995.
[11] The Telecommunications Industry Association (TIA). *Mobile Station-Base Station Compatibility Standard for Wideband Spread Spectrum Cellular Systems (TIA/EIA-95-B-99)*, Feb 1999.
[12] V. Varadharajan and Y. Mu. Preserving privacy in mobile communications: A hybrid method. In *IEEE International Conference on Personal Wireless Communications*, pages 532–536, 1997.