Server Impersonation Attacks on RFID Protocols

Boyeon Song
Information Security Group
Royal Holloway, University of London
Egham, Surrey, TW20 0EX, UK
b.song@rhul.ac.uk

Abstract

We introduce server impersonation attacks, a practical security threat to RFID security protocols that has not previously been described. RFID tag memory is generally not tamper-proof for cost reasons. We show that, if a tag is compromised, such attacks can give rise to desynchronisation between server and tag in a number of existing RFID authentication protocols. We also describe possible countermeasures to this novel class of attacks.

1 Introduction

Radio Frequency IDentification (RFID) is a means for identifying objects via a radio signal, and enables automated data gathering in a variety of applications, from entry access controls to electronic passports [4, 5]. An RFID device, also known as an RFID tag, is a small integrated circuit with a unique identifier which transmits data over the air in response to interrogation by an RFID reader [4]. RFID devices are being deployed in a range of applications, including door entry cards, transportation payment cards, and novel forms of credit cards, and in the near future they are likely to be carried by many people as a means of identification, e.g. as a national ID card or electronic passport [4, 5].

RFID devices can be divided into two broad types, 'dumb' and 'smart'. A dumb tag has no significant processing power, and its unique identifier will normally be a fixed length value, typically 10 or 16 hexadecimal digits long, and its memory capacity is likely to be fairly low — of the order of a few hundred bytes to a maximum of around 2kBytes [5]. A smart tag, by contrast, has on-board processors that are typically capable of performing cryptographic operations [5]. It will often have a much larger memory capacity of 32kBytes or more, and will be able to require authentication before allowing access to stored data [5]. Such a tag may also be able to encrypt communications using session keys to avoid snooping or data injection attacks [5].

Our interest here is in smart RFID tags, and we focus in particular on the means of authentication used to access tag-specific information stored in a back-end server. A variety of security and privacy threats to RFID authentication protocols have been widely studied, including eavesdropping, replay attacks, denial of service (DoS) attacks, tracking, and traceability. In this paper, we introduce another practical threat, namely server impersonation attacks. If the contents of tag memory are revealed to an attacker, then such an attack could be used to cause desynchronisation between a back-end server and a tag. Tag memory compromise is a genuine practical threat since RFID tags tend to be susceptible to physical attacks, given that tag memory is not likely to be tamper-resistant for cost reasons. In other words, server impersonation is a realistic attack on RFID systems, which should be considered when evaluating any candidate RFID protocol.

We first give a general description of attack models and present server impersonation attacks on synchronisation-based RFID protocols. Section 3 describes how a server impersonation attack can bring about desynchronisation in a number of existing RFID schemes. We then propose possible countermeasures to server impersonation attacks; finally, we summarise our results.

2 Preliminaries

2.1 RFID Protocols

Many protocols have been proposed for use in RFID systems (see, for example, [1, 2, 3, 4, 6, 7, 8, 10]). We focus here on RFID authentication protocols requiring synchronisation between a tag and a back-end server, which operate under the following assumptions.

- An RFID system incorporates components of two types, namely a back-end server and RFID tags.

- Each server maintains a server database (DB) containing a set of values for each tag that it manages, and is combined with an RFID reader.
- Each tag has a rewritable memory which may be susceptible to compromise.
- The channel between the server and the tag is insecure, and communications are subject to eavesdropping or modification.
- An RFID protocol consists of three flows; typically, the first flow is a query from a server to a tag, the second is the reply of the tag to the server for tag authentication, and the third is the response from the server to the tag for server authentication.
- A server and a tag share secrets used for mutual authentication. They update the shared secrets synchronously whenever they perform a successful authentication session; a server updates tag secrets stored in its DB after receiving the second flow and having authenticated the tag, and the tag updates its stored secrets after receiving the third flow and having authenticated the server.

2.2 Attack Models

Security threats to RFID protocols can be classified into weak and strong attacks. Weak attacks are threats feasible just by observing and manipulating communications between a server and tags. Replay attacks and interleaving attacks are examples of weak attacks.

Strong attacks are threats possible for an attacker which has compromised a target tag. An RFID tag's memory is vulnerable to compromise by side channel attacks, because the memory of a low cost tag is unlikely to be tamper-proof. Hence, strong as well as weak attacks should be considered in RFID protocol design. Backward traceability, forward traceability, and server impersonation attacks, as described below, are all examples of strong attacks.

Suppose that an attacker compromises a target tag at time t. This attacker might also have intercepted tag interactions that occurred at a time $t' < t$ [6]. The past transcripts of the tag, when combined with the information gained from the tag compromise, might allow tracking of the tag owner's past behaviour [6]. Such an attack is called here *backward traceability* (resistance to such an attack is sometimes also referred to as *forward security*).

Also, an adversary that has all the internal state of a target tag at time t might be able to identify the interactions of this tag that occur at a future time $t' > t$ [6]. Such an attack, called *forward traceability*, is related to tag ownership transfer. This is because, if an RFID scheme does not provide forward untraceability, when the ownership of a tag is transferred, the previous owners might be able to read communications between the new owner and the tag.

In addition to these traceability threats, an adversary that has compromised a tag could impersonate a valid server using knowledge of the tag's internal state. Such an adversary might be able to ask the tag to update its internal state, with the effect that the tag can no longer communicate successfully with the real server. Such a *server impersonation attack* is a significant issue for secure tag ownership transfer, as well as in relation to backward and forward traceability. For example, suppose that an RFID protocol does not resist forward traceability and server impersonation. Suppose further that, using this protocol, a previous tag owner has passed ownership of a tag to a new owner, but knows the tag secrets at the time of transfer. The previous owner might be able to use this knowledge to impersonate the new owner's server to the tag, and change the tag secrets after ownership transfer. As a result, the new owner might no longer be able to read the tag successfully, and only the previous owner would be able to identify the tag. This attack does not appear to have been discussed previously, despite its potential importance. We discuss such server impersonation attacks in greater detail below.

2.3 Server Impersonation Attacks

Server impersonation means that an adversary is able to impersonate a valid server to a tag. One reason that this is a genuine threat is because desynchronisation can occur if a tag updates its stored data when the server does not. More specifically, in protocols satisfying the assumptions given in section 2.1, an attacker that has read a tag's stored secrets could impersonate an authorised server to the tag. If the attacker executes an authentication session with the tag, impersonating a valid server, then it could make the tag update its stored secrets, although the genuine server will not update its DB entry. The tag and the real server would then be desynchronised, and incapable of successful communications.

Whether or not the compromise of tag stored secrets enables such a server impersonation attack is the main focus of the next section.

3 Server Impersonation Attacks on RFID Protocols

Our focus here is on desynchronisation attacks arising from server impersonation attacks, as discussed in section 2.3. We now review four recently proposed RFID schemes as typical examples of protocols fitting the model given in section 2.1; in each case we consider whether or not the compromise of a tag's secret data enables a server impersonation attack.

The following notation is used throughout this section.

T_i	The i-th tag ($1 \leq i \leq$ the number of tags)
$\hat{x}$	The most recent value of x (for any value x)
$\overline{x}$	The updated value of x (for any value x)
r	A random number
$\oplus$	XOR operator
$\|$	Concatenation operator
$\leftarrow$	Substitution operator

3.1 The Henrici-Müller protocol

This scheme (referred to here as the HM protocol) was proposed by Henrici and Müller in 2004 [3].

3.1.1 Protocol Description

The HM scheme uses a one-way hash function h and a binary operation $\circ$ on bit strings (a simple exclusive-or function is adequate for this purpose [3]). A server DB contains a table with the following entries for each tag T_i: the hash of the current tag identifier HID_i that acts as the primary index for the table, the current tag identifier ID_i, a transaction number TID_i, the number of the last successful transaction LST_i, and the DB entry for the associated tag AE_i. A tag T_i stores the values of ID_i, TID_i and LST_i. If the authentication process completes correctly, the tag and the server will update their stored copies of ID_i to the value $ID_i \circ r$. Figure 1(a) summarises the HM protocol, where time flows from the top to the bottom, i.e. the top-most message is sent first and the bottom-most message is sent last. The other figures in this paper use the same time convention.

3.1.2 Server Impersonation Attack

It is straightforward for a strong attacker to carry out a server impersonation based desynchronisation attack on the HM protocol. If an attacker knows the secrets stored in tag memory, i.e. ID_i, TID_i and LST_i, then the attacker can impersonate the server and complete a successful authentication session with the tag. The attacker can send a query to the target tag, receive M_1, M_2 and M_3 from the tag in reply, and then send a valid response, r and M_4, to the tag. The tag will then verify M_4, and update its stored value of ID_i to $ID_i \circ r$, but the server will not update its value of ID_i. As a result, the server and the tag will become desynchronised. Figure 1(b) summarises this server impersonation attack.

3.2 The Dimitriou protocol

In 2005, Dimitriou [2] proposed an RFID authentication protocol (referred to here as the D protocol) to enforce user privacy and protect against tag cloning.

Server $[HID_i, ID_i, TID_i, LST_i, AE_i]$		**Tag** T_i $[ID_i, TID_i, LST_i]$
	$\overset{Query}{- - - \rightarrow}$	
		$TID_i \leftarrow TID_i + 1$
		$M_1 = h(ID_i)$
		$M_2 = h(ID_i \circ TID_i)$
		$M_3 = TID_i - LST_i$
	$\overset{M_1, M_2, M_3}{\leftarrow - - -}$	
Search for HID_i in the DB for which $HID_i = M_1$		
$TID'_i \leftarrow LST_i + M_3$		
If $M_2 = h(ID_i \circ TID'_i)$ & $TID'_i > TID_i$		
Generate r		
$M_4 = h(r \circ TID'_i \circ ID_i)$		
	$\overset{r, M_4}{- - - \rightarrow}$	Verify
Update		$M_4 \overset{?}{=} h(r \circ TID_i \circ ID_i)$
$AE_i \leftarrow ID_i$		
$ID_i \leftarrow ID_i \circ r$		If OK, update
$HID_i \leftarrow h(ID_i)$		$ID_i \leftarrow ID_i \circ r$
$TID_i \leftarrow TID'_i$		$LST_i \leftarrow TID_i$
$LST_i \leftarrow TID'_i$		

(a) The HM Protocol

Adversary $[ID_i, TID_i, LST_i]$		**Tag** T_i $[ID_i, TID_i, LST_i]$
	$\overset{Query}{- - - \rightarrow}$	$\overline{TID_i} \leftarrow TID_i + 1$
	$\overset{M_1, M_2, M_3}{\leftarrow - - -}$	
	$\overset{r, M_4}{- - - \rightarrow}$	$\overline{ID_i} \leftarrow ID_i \circ r$
		$\overline{LST_i} \leftarrow TID_i$
(After the attack)		
Server $[HID_i, ID_i, TID_i, LST_i, AE_i]$		**Tag** T_i $[\overline{ID_i}, \overline{TID_i}, \overline{LST_i}]$

(b) Server Impersonation Attack on the HM Protocol

Figure 1. Server Impersonation Attack on the HM Protocol

3.2.1 Protocol Description

In the D scheme, a tag T_i stores its identifier ID_i, and a server DB stores the identifier ID_i and a hash of the identifier HID_i for each tag T_i, where HID_i serves as the primary key used to identify information related to the tag. This scheme uses a hash function h and a keyed hash function h_k. When a session ends successfully, the copies of the identifier ID_i held by the tag and the server are updated using an irreversible function, denoted by f. The D protocol is summarised in Figure 2(a).

3.2.2 Server Impersonation Attack

The D scheme is subject to a server impersonation based attack analogous to that described on the HM protocol. If an adversary knows ID_i, then the adversary can impersonate the server to conduct an authentication session with T_i. The

adversary will receive r_2, M_1 and M_2 as a response from T_i when it sends a query r_1 to the tag. Using the compromised tag information, the adversary can then respond with a valid M_3. As a result of receiving M_3, the tag will update its copy of the identifer ID_i to $f(ID_i)$; however, the server DB will not be updated. The server and the tag will then become desynchronised. Figure 2(b) summarises this server impersonation attack.

Server $[HID_i, ID_i]$		**Tag** T_i $[ID_i]$
	$\overset{r_1}{- - - \rightarrow}$	
		Generate r_2 $M_1 = h(ID_i)$ $M_2 = h_{ID_i}(r_2, r_1)$
	$\overset{r_2, M_1, M_2}{\leftarrow - - -}$	
Search for HID_i in the DB for which $HID_i = M_1$		
If $M_2 = h_{ID_i}(r_2, r_1)$ $ID_i \leftarrow f(ID_i)$ $M_3 = h_{ID_i}(r_2, r_1)$		
	$\overset{M_3}{- - - \rightarrow}$	Verify $ID_i' \leftarrow f(ID_i)$ $M_3 \overset{?}{=} h_{ID_i'}(r_2, r_1)$
		If OK, update $ID_i \leftarrow ID_i'$

(a) The D Protocol

Adversary $[ID_i]$		**Tag** T_i $[ID_i]$
	$\overset{r_1}{- - - \rightarrow}$	
	$\overset{r_2, M_1, M_2}{\leftarrow - - -}$	
	$\overset{M_3}{- - - \rightarrow}$	$\overline{ID}_i \leftarrow f(ID_i)$
(After the attack).....		
Server $[HID_i, ID_i]$		**Tag** T_i $[\overline{ID}_i]$

(b) Server Impersonation Attack on the D Protocol

Figure 2. Server Impersonation Attack on the D Protocol

3.3 The Chien-Chen protocol

A mutual authentication protocol for use in RFID systems was proposed by Chien and Chen in 2007 [1] (we refer to this as the CC protocol). It is based on the EPC Class 1 GEN-2 standard.

3.3.1 Protocol Description

A server initially stores three values in a tag T_i, namely the Electronic Product Code (EPC) of the tag EPC_i, the initial authentication key K_i, and the initial access key P_i. The values of K_i and P_i are updated after each successful authentication. The server also maintains five values in its DB for each tag T_i: EPC_i, the new authentication key K_i, the new access key P_i, the most recent old authentication key $\hat{K}_i$, and the most recent old access key $\hat{P}_i$. It uses simple cryptographic primitives such as a Pseudo-Random Number Generator (PRNG) and a Cyclic Redundancy Code (CRC). We use f and h respectively to denote the PRNG and CRC functions used by the scheme. Figure 3(a) summarises the CC protocol.

3.3.2 Server Impersonation Attack

The CC scheme is designed to resist replay attacks, DoS attacks and backward traceability [1]. However, server impersonation attacks remain a practical threat to synchronisation between the server and the tag.

An adversary that has read EPC_i, K_i and P_i from a tag T_i can commence a session with the tag by sending a random number r_1. When the tag responds with r_2 and M_1, the attacker is able to send the expected value of M_2 back to the tag. As a result, the tag will update both its session key K_i and its access key P_i. The tag will then have stored keys that are different to those in the server DB, as shown in Figure 3(b).

3.4 The Song-Mitchell protocol

Song and Mitchell proposed an RFID authentication protocol (referred to here as the SM protocol) in 2008 [9].

3.4.1 Protocol Description

A server DB stores secrets u_i and t_i for each tag T_i as well as the most recent secrets $\hat{u}_i$ and $\hat{t}_i$. Initially, secret u_i is a string of l bits assigned to T_i, and t_i is a hash of u_i, i.e. $t_i = h(u_i)$. A tag T_i stores the value of t_i as its identifier. This scheme uses a hash function h to update the secret t_i, a keyed hash function f to protect messages, and a combination of simple functions such as right and left shifts ($\gg, \ll$) and a bit-wise exclusive-or operation ($\oplus$) to combine data strings. When an authentication session completes successfully, the server DB updates the secret values of (u_i, t_i) and ($\hat{u}_i$, $\hat{t}_i$) for T_i, and the tag also updates its secret t_i using h. Figure 4 summarises the SM protocol.

3.4.2 Server Impersonation Attack

In the SM scheme, the value u_i that is used to authenticate the server to the tag in the third flow of a session is known to the server but not the tag. This means that an adversary that has compromised a tag is not able to impersonate a server to the tag using an approach analogous to those described above for the other three protocols. That is, an adversary

Server		**Tag** T_i
$[EPC_i, K_i, P_i, \hat{K}_i, \hat{P}_i]$		$[EPC_i, K_i, P_i]$
	$\overset{r_1}{- - - \rightarrow}$	
		Generate r_2 $M_1 = h(EPC_i \parallel r_1 \parallel r_2) \oplus K_i$
	$\overset{r_2, M_1}{\leftarrow - - -}$	
Search for EPC_i in the DB for which $M_1 \oplus K_i = h(EPC_i \parallel r_1 \parallel r_2)$		
Compute $M_2 = h(EPC_i \parallel r_2) \oplus P_i$		
	$\overset{M_2}{- - - \rightarrow}$	Verify
Update		$M_2 \oplus P_i \overset{?}{=} h(EPC_i \parallel r_2)$
$\hat{K}_i \leftarrow K_i$		
$\hat{P}_i \leftarrow P_i$		If OK, update
$K_i \leftarrow f(K_i)$		$K_i \leftarrow f(K_i)$
$P_i \leftarrow f(P_i)$		$P_i \leftarrow f(P_i)$

(a) The CC Protocol

Adversary		**Tag** T_i
$[EPC_i, K_i, P_i]$		$[EPC_i, K_i, P_i]$
	$\overset{r_1}{- - - \rightarrow}$	
	$\overset{r_2, M_1}{\leftarrow - - -}$	
	$\overset{M_2}{- - - \rightarrow}$	$\overline{K}_i \leftarrow f(K_i)$ $\overline{P}_i \leftarrow f(P_i)$
(After the attack)		
Server		**Tag** T_i
$[EPC_i, K_i, P_i, \hat{K}_i, \hat{P}_i]$		$[EPC_i, \overline{K}_i, \overline{P}_i]$

(b) Server Impersonation Attack on the CC Protocol

Figure 3. Server Impersonation Attack on the CC Protocol

Server		**Tag:** T_i
$[u_i, t_i, \hat{u}_i, \hat{t}_i]$		$[t_i]$
	$\overset{r_1}{- - - \rightarrow}$	
		Generate r_2 $M_1 = t_i \oplus r_2$ $M_2 = f_{t_i}(r_1 \oplus r_2)$
	$\overset{M_1, M_2}{\leftarrow - - -}$	
Search for t_i in the DB for which $r_2 \leftarrow M_1 \oplus t_i$ and $M_2 = f_{t_i}(r_1 \oplus r_2)$		
Compute $M_3 = u_i \oplus (r_2 \gg l/2))$		
	$\overset{M_3}{- - - \rightarrow}$	Verify
Update		$u_i \leftarrow M_3 \oplus (r_2 \gg l/2)$
$\hat{u}_i \leftarrow u_i$		$t_i \overset{?}{=} h(u_i)$
$\hat{t}_i \leftarrow t_i$		
$u_i \leftarrow (u_i \ll l/4) \oplus (t_i \gg l/4) \oplus r_1 \oplus r_2$		If OK, update $t_i \leftarrow h((u_i \ll l/4) \oplus (t_i \gg l/4) \oplus r_1 \oplus r_2)$
$t_i \leftarrow h(u_i)$		

Figure 4. SM protocol

cannot masquerade as a valid server to a tag T_i even if it knows the tag secret t_i, because it does not know u_i and hence is unable to compute M_3.

However, a server impersonation attack could still be feasible if an adversary performs a more elaborate attack; if an adversary first performs a DoS attack or a tag impersonation attack in a previous valid authentication session, and obtains u_i from the messages sent in this session (using the compromised tag secret t_i), it could subsequently carry out a server impersonation attack in a session with the tag, as long as no valid session has been performed since the tag compromise. However, such an attack is rather complex, and may be very difficult to conduct in practice.

4 Countermeasures and Future Work

Once an RFID tag's stored secrets have been compromised, it is difficult to prevent server impersonation based desynchronisation attacks, as we have shown in our analysis of four existing RFID protocols in section 3.

We could attempt to design more robust RFID protocols that make such server impersonation attacks more difficult to perform. If an RFID protocol uses a digital signature scheme for authentication of a server to a tag, then a strong attacker is unable to impersonate the server to a tag just by compromising the tag. However, the use of public key cryptography may be beyond the capabilities of many tags.

Another possible countermeasure involves a tag T_i storing the most recent old secrets as well as its current secrets, as proposed for the server DB. As a result, the protocol would be more resistant to a desynchronisation threat caused by a server impersonation attack; an adversary would have to carry out at least two sessions to succeed in such a desynchronisation attack. However, such an approach might increase tag cost.

A further possible alternative approach would be to require the server to possess a secret that is used for server authentication to a tag and that is not known to the tag, like the value u_i in the SM protocol. In this latter protocol, the server DB has secrets u_i and t_i, and the tag has only one secret t_i. The tag can authenticate the server by checking that the value of u_i sent by the server is valid, because u_i is a hash of t_i. Thus, such a scheme can resist server impersonation attacks resulting from tag compromise. As described in section 3.4.2, a server impersonation based desynchronisation attack on the SM scheme remains possible, but the method is rather complex. This may be sufficient to make such attacks impractical.

An important challenge for future work is to design a robust and practical RFID protocol that is able to resist strong attacks such as server impersonation, whilst minimising cost and maximising performance.

5 Concluding remarks

A server impersonation based desynchronisation attack is a feasible security threat because RFID tag memory is typically not tamper-resistant. Moreover, in some cases, the ownership of a tag may change. We have shown how, in cases where tag memory has been compromised, certain previously proposed RFID protocols can be desynchronised by a server impersonation attack; such an attack is relatively straightforward to perform on the MH, D and CC schemes, and more difficult for the SM scheme, because of its use of an authentication key known to the server but not the tag. We have also proposed possible countermeasures designed to make an RFID protocol more resistant to such server impersonation attacks — one problem is that implementing these measures might increase tag cost. That is, we have a trade-off between security and cost. We conclude that server impersonation attacks should be considered in any future security assessment of an RFID protocol.

Acknowledgements

I am very appreciative of all the comments and encouragement that my supervisor, Chris J. Mitchell, has given me.

References

[1] H. Chien and C. Chen. Mutual authentication protocol for RFID conforming to EPC class 1 generation 2 standards. *Computer Standards & Interfaces*, 29(2):254–259, February 2007.

[2] T. Dimitriou. A lightweight RFID protocol to protect against traceability and cloning attacks. In *Conference on Security and Privacy for Emerging Areas in Communication Networks — SecureComm*, pages 59–66, Athens, Greece, September 2005. IEEE.

[3] A. Henrici and P. Müller. Hash-based enhancement of location privacy for radio-frequency identification devices using varying identifiers. In R. Sandhu and R. Thomas, editors, *International Workshop on Pervasive Computing and Communication Security — PerSec 2004*, pages 149–153, Orlando, Florida, USA, March 2004. IEEE Computer Society.

[4] A. Juels. RFID security and privacy: A research survey. *IEEE Journal on Selected Areas in Communications*, 24:381–394, February 2006.

[5] A. Laurie. Practical attacks against RFID. *Network Security*, 2007(9):4–7, September 2007.

[6] C. Lim and T. Kwon. Strong and robust RFID authentication enabling perfect ownership transfer. In P. Ning, S. Qing, and N. Li, editors, *Conference on Information and Communications Security — ICICS '06*, volume 4307 of *Lecture Notes in Computer Science*, pages 1–20, Raleigh, North Carolina, USA, December 2006. Springer-Verlag.

[7] D. Molnar and D. Wagner. Privacy and security in library RFID: Issues, practices, and architectures. In B. Pfitzmann and P. Liu, editors, *Conference on Computer and Communications Security — ACM CCS*, pages 210–219, Washington, DC, USA, October 2004. ACM Press.

[8] M. Ohkubo, K. Suzki, and S. Kinoshita. Cryptographic approach to "privacy-friendly" tags. In *RFID Privacy Workshop*, MIT, MA, USA, November 2003. http://www.rfidprivacy.us/2003/agenda.php.

[9] B. Song and C. J. Mitchell. RFID authentication protocol for low-cost tags. In V. D. Gligor, J. Hubaux, and R. Poovendran, editors, *ACM Conference on Wireless Network Security — WiSec '08*, pages 140–147, Alexandria, Virginia, USA, April 2008. ACM press.

[10] S. Weis, S. Sarma, R. Rivest, and D. Engels. Security and privacy aspects of low-cost radio frequency identification systems. In D. Hutter, G. Müller, W. Stephan, and M. Ullmann, editors, *International Conference on Security in Pervasive Computing — SPC 2003*, volume 2802 of *Lecture Notes in Computer Science*, pages 454–469, Boppard, Germany, March 2003. Springer-Verlag.