3GPP TSG SA WG3 Security — S3#32 **S3-040047**
09 - 13 February 2004
Edinburgh, Scotland, UK

Source:	**Ericsson**
Title:	**Replay attacks in the split UE scenario**
Document for:	**Discussion**
Agenda Item:	**TBA**

1. Introduction

In [1], Eric Gauthier describes a replay attack where a man-in-the-middle captures Bluetooth communications and simultaneously uses a compromised Wireless LAN access point to trick a laptop into believing it has successfully authenticated with a 3G-WLAN network.

The conditions necessary for this attack to succeed have been discussed on the SA3 mailing list. It turns out that the conditions are dependent on the protection over the Bluetooth interface, which cryptographic quantity is calculated by the terminal, and what kind of EAP method is run from the laptop. This paper describes those conditions.

2. Attack

Eric's attack can be be summarized as follows:

1. Attacker records Bluetooth communications between a terminal and a laptop related to the EAP-SIM or EAP-AKA exchange. These communications are encrypted, and it is not assumed that the attacker can recover the encryption keys.
2. Simultaneously, attacker recovers the Master Session Key (MSK) related to this session from the access point. It is assumed that the access point has been previously compromised.
3. The attacker can now replay both the terminal - laptop and access point – laptop communications against the same client at a different location and/or at a later time.
4. The victim, the client, believes it has correctly attached to a WLAN network using cellular authentication. In reality, it is attached to the attacker's access point.

3. Conditions for the Attack

The attack works on the Bluetooth interface because the laptop does not introduce randomness to the used key, and because there is no replay and integrity protection in the used Bluetooth communications service. The attack works on the WLAN interface because the laptop does not introduce randomness that would invalidate the replayed authentication exchange. In the following we study the latter situation in more detail.

Some concepts are defined first, quoting from [2]:

EAP Master key (MK)

A key derived between the EAP client and server during the EAP authentication process, and that is kept local to the EAP method and not exported or made available to a third party.

Master Session Key (MSK)

Keying material (at least 64 octets) that is derived between the EAP client and server and exported by the EAP method.

Transient EAP Keys (TEKs)

Session keys which are used to establish a protected channel between the EAP peer and server during the EAP authentication exchange. The TEKs are appropriate for use with the cipher suite negotiated between EAP peer and server for use in protecting the EAP conversation. Note that the cipher suite used to set up the protected channel between the EAP peer and server during EAP authentication is unrelated to the cipher suite used to subsequently protect data sent between the EAP peer and authenticator. An example TEK key hierarchy is described in Appendix C of [2].

Different realizations of Eric's attack exist, based on what kind of functional split exists between the terminal and the laptop, and the type of EAP method used. All of them have some common criteria, however. The success of the attack requires that (1) the Bluetooth communications are affected only by quantities chosen by the attacker from the terminal, and (2) the EAP communications are affected only by quantities chosen by the attacker from the access point.

Lets look at some specific protocol designs and see how they are affected by Eric's attack:

1. Delivery of the MK versus the MSK to the laptop.

   The MSK is the end-result of the EAP authentication, and typically available only at the end of the EAP exchange. This implies that no EAP message uses the MSK. Consequently, if the terminal is responsible for generating the MSK and the processing of all messages prior to it, the laptop has no EAP means to determine whether the MSK is correct. Or if it is correct but replayed from an older authentication run.

   The situation is different if the terminal is responsible for generating the MK, and the laptop is responsible for using the MK in subsequent EAP messaging. Then there is a possibility for the laptop to detect, e.g., replays. However, it turns out that this is not a sufficient condition as is discussed below.

2. Specific EAP method exchange.

   Even if the terminal delivers just the MK to the laptop, this may not help unless the part of the EAP method that was left for the laptop has some specific properties. In particular, it is necessary for there to be some replay protection or randomness introduced by the laptop in this remaining part. For instance, in EAP-AKA the replay protection is provided entirely by the AKA procedure itself, and if that is delegated to the terminal in a way that enables replays, there is nothing else that the laptop can do.

   In EAP-SIM, the laptop introduces a nonce, so the situation is different. However, it this turns out to be insufficient, as discussed below.

3. Ability to divide work.

   EAP-SIM calculates the MK as follows (omitting unnecessary details):

   MK = PRF (… Kc … nonce …)

   One of the reasons for the division of work between the laptop and the terminal is the hiding of Kc values from an insecure environment. Thus, we should find a way for the division of work to happen so that the terminal is responsible for calculating the part that hides the Kc, and the laptop is responsible for including the nonce. However, the above formula does not make this easy, as all parameters are included in the same PRF invocation.

5. Conclusions

A secure protocol can be made insecure by distributing the execution of its parts on different parties, particularly if the underlying communications are not protected well enough.

Integrity and replay protection over the Bluetooth interface is necessary in this application.

6. References

[1] Eric Gauthier. A man-in-the-middle attack using Bluetooth in a WLAN interworking environment. December, 2003.
[2] B. Aboba et al. EAP Keying Framework. Internet Draft draft-ietf-eap-keying-01.txt, November 2003.