Symbolic Analysis for Security of Roaming Protocols in Mobile Networks [Extended Abstract]

Chunyu Tang, David A. Naumann*, and Susanne Wetzel

Stevens Institute of Technology, Hoboken NJ 07030 USA

Abstract. Both GSM (2G) and UMTS (3G) wireless standards are deployed worldwide. Like the 4G standard now appearing, these standards provide for mobile devices with differing capabilities to roam between providers or technologies. This poses serious challenges in ensuring authentication and other security properties. Automated analysis of security properties is needed to cope with the large number of possible scenarios. While some attacks exploit weaknesses in cryptographic functions, many attacks exploit flaws or other features of the protocol design. The latter attacks can be found using symbolic (Dolev-Yao) models. This paper demonstrates the use of a fully automatic tool to exhaustively analyze symbolic models of GSM, UMTS, and the respective roaming protocols. The results include the demonstration of known attacks as well as the confirmation of expected properties.

Keywords: Mobile networks, GSM, UMTS, roaming protocols, security, authentication, secrecy, symbolic modeling, automated analysis.

1 Introduction

Starting in the early 90s, the Global System for Mobile Communications (GSM)—also referred to as second generation (2G) technology—became a main standard for mobile telecommunication. Recently, phone carriers have started to build fourth generation (4G) networks. Given the evolution of the telecommunication standards and devices, enabling secure roaming and handover is at the core of providing service to mobile subscribers.[1] This includes roaming and handover within one technology as well as across technologies.

While Universal Mobile Telecommunications System (UMTS)—also called third generation (3G)—allows for the interoperation with GSM, its successor is to support many more options. Analyzing the security of all possible combinations of interoperating scenarios by hand is a rather tedious undertaking. While for the interoperation between GSM and UMTS six different scenarios had to be investigated, this number grows an order of magnitude for 4G.

* Tang and Naumann were supported by US NSF award CCF-0915611.

[1] The term *handover* refers to mechanisms for mobility of an ongoing phone call or data exchange; *roaming* refers to mobility while no such service is currently active.

M. Rajarajan et al. (Eds.): SecureComm 2011, LNICST 96, pp. 480–490, 2012.

This paper explores an approach that allows for a structured exploration of specified security properties in the various contexts. The first contribution is to provide formal models of the main security components of both the GSM and UMTS standards (Sects. 3 and 4), together with the specification of required authenticity and secrecy properties. The second contribution is to show that one can automate the checking of the various (roaming) scenarios to see whether or not specified security properties hold—thus replacing the tedious checking of all possible combinations by hand (Sect. 5). The automated analysis finds several known attacks, and does not miss known attacks except those that exploit weaknesses in cryptographic functions. The feasibility study in this paper lays the basis for a formal modeling and automated analysis of 4G and its manifold associated roaming and handover scenarios.

2 Related Work

The security of the mobile communication standards has been studied extensively both in the context of the standardization efforts [19] and outside by the greater research and user community. For example, [8] noted that GSM is susceptible to a false base station attack. Recently, Nohl et al. demonstrated that it is possible to eavesdrop on GSM communication in practice with rather little effort. Attacks on GSM ciphers were devised by, for example, Golic [9], and Dunkelman et al. [6]—the latter of which is also of relevance in the context of UMTS security. Aside from general assessments of UMTS security (e.g., [16]), or works focusing on UMTS encryption and integrity mechanisms (e.g., [16,11]), one main focus in UMTS is on security in the context of roaming and handover (e.g., [14,15,13]).

In this paper we focus on the analysis of protocols in the *symbolic* or Dolev-Yao model, which assumes perfect cryptography: an encrypted message can be decrypted only with the appropriate key, hash functions do not have collisions, etc. The attacker has complete control over the network: it can introduce, alter, replay, and delete any message. Several open-source protocol analyzers have seen extensive use; we mention selected examples. Taha et al. [17] analyze the handover schemes and the pre-authentication handover protocol in IEEE 802.16e standard (Mobile WiMAX) using the Scyther tool. They find the attacker can obtain the keys in both protocols. Taha et al. [18] analyze the privacy and key management protocol in IEEE 802.16 (PKMv1) and 802.16e (PKMv2) using Scyther. They find a pseudonymity attack in both protocols, and an attack violating the data secrecy in PKMv1. Lim et al. [12] propose a handover protocol for WLAN, WiBro and UMTS, and verify it using AVISPA. Bouassida et al. [4] analyze the key management architecture for hierarchical group protocols using AVISPA, and find an attack on the members promotion protocol. Other relevant tools are LySa [3] and protocol analyzers of Meadows et al. [7].

In this work we use the ProVerif tool (PV for short). Chang and Shmatikov [5] use PV to analyze the Bluetooth device pairing protocols; They confirm the offline guessing attack [10] and discover an attack scenario for a then-proposed

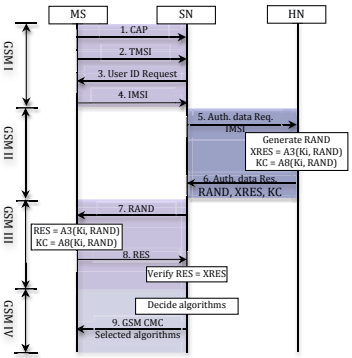

Fig. 1. GSM message sequence diagram [19, 13]

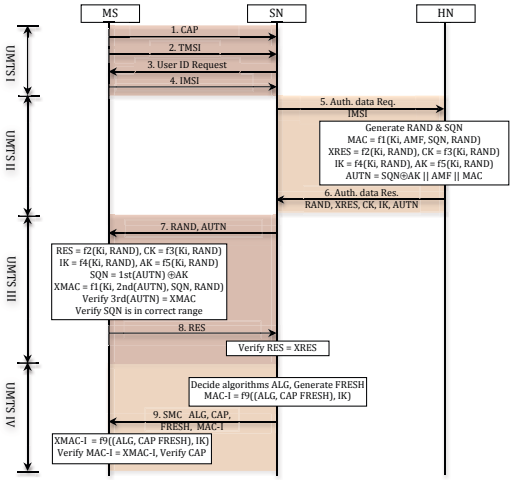

Fig. 2. UMTS message sequence diagram [19, 13]

Simple Pairing protocol. Abadi et al. use PV to analyze the Just Fast Key protocol [1]. Blanchet and Chaudhuri [2] use PV to analyze a protocol for secure file sharing on untrusted storage.

3 Review of GSM, UMTS, and Roaming

3.1 GSM

When connecting to a *Base Station* (BS) of a *Servicing Network* (SN), a *Mobile Station* (MS) first transmits its *Temporary Mobile Subscriber Identity* (TMSI) as well as its *CAPabilities* (CAP), i.e., the encryption algorithms it supports to the SN. In case the SN does not recognize the TMSI (i.e., the SN cannot resolve the received TMSI to a unique *International Mobile Subscriber Identity* (IMSI), the SN will require the MS to send its unique IMSI. A SN may be the MS's wireless service provider, i.e., its *Home Network* (HN) or a *Foreign Network* (FN). A MS might receive service from a FN in areas where its HN does not provide service. Based on the IMSI, the SN will then request an authentication vector from the MS's HN (see Message 5 in Fig. 1).

The MS and HN share a long-term symmetric secret key K_i. Using this pre-shared secret key K_i and a random nonce $RAND$, the HN generates an authentication vector consisting of the three components $RAND$, $XRES$, and a session key K_c. (The details of the algorithms A3 and A8 [19] which are used to compute the $XRES$ and K_c are not important in the context of this paper.) Upon receiving the authentication vector (Message 6), the SN starts a challenge-response type exchange with the MS. Specifically, the SN sends the challenge $RAND$ to the MS (Message 7). Based on the received $RAND$ and the pre-shared key K_i stored on its SIM card, the MS computes the session key K_c as well as a response RES to the received challenge $RAND$. Subsequently, the MS sends the

computed response *RES* to the SN. If *RES* matches the response *XRES* (which the SN received from the HN as part of the authentication vector) then the MS has successfully authenticated itself to the SN. Then the SN moves on to decide which encryption algorithm will be used to encrypt the communication on the air interface between the SN and the MS (based on the CAP it received from the MS earlier). The SN informs the MS of its choice (Message 9) and starts the (possibly encrypted) communication.

Security in GSM: The protocol described above assures the SN of the *authenticity* of the MS in case the response *RES* received from the MS matches *XRES* received from the HN. Only an MS holding the pre-shared key K_i will be able to compute the correct result *RES* in response to the random challenge *RAND* (assuming the security of algorithms A3 and A8). However, GSM does not provide any assurance for the MS that it is in fact connected to a legitimate BS of the SN. The *false base station attack* is a *man-in-the-middle attack* [8], i.e., the false BS sits in between the MS and the legitimate BS, pretending to be a legitimate BS when communication with the MS and pretending to be a legitimate MS when communicating with the actual BS. Since the MS sends its capabilities CAP in the clear, it is possible for a false BS to intercept and modify that information before forwarding it to a legitimate BS. In particular, it can change the capabilities CAP it received from the MS so that the legitimate BS will believe that the MS cannot support encryption. Consequently, the false BS will be able to listen in and arbitrarily modify the unencrypted communication between the MS and the legitimate BS.

3.2 UMTS

Establishing a connection to a BS of a SN in UMTS follows the same principles as those in GSM. The main differences lie in that those specified in UMTS are intended to provide mutual authentication of the MS and the SN and they also include mechanisms to support integrity protection and to establish freshness guarantees. Specifically, upon receiving an authentication request from the SN (Message 5 in Fig. 2), the HN computes an authentication vector that is more extensive than the one issued in GSM. In addition to determining a challenge response pair *RAND* and *XRES*, in UMTS the HN also computes a *Message Authentication Code* (MAC), a *Ciphering Key* (CK), an *Integrity Protection Key* (IK), an *Anonymity Key* (AK), a *Sequence Number* (SQN) as well as an *Authentication Management Field* (AMF).

Upon receiving the authentication vector (Message 6), the SN initiates the challenge response exchange with the MS. Unlike in GSM, the MS receives the *Authentication Token* (AUTN) as part of Message 7 which allows the MS to determine whether or not this is a *fresh* authentication challenge. Upon receiving the response *RES* from the MS (Message 8), the SN checks whether or not it matches *XRES*. If so, then the MS has successfully authenticated to the SN. The SN then proceeds to deciding on the encryption algorithm. In contrast to GSM, in UMTS the SN includes the CAP (MS's encryption and integrity protection capabilities) in its message to the MS as it received them as part of Message 1.

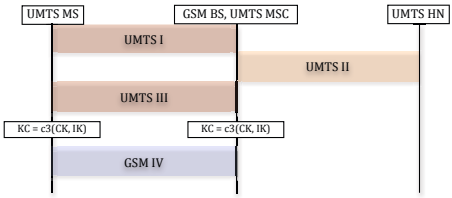

Fig. 3. UMTS subscriber roams into mixed network (blocks GSM I–IV, UMTS I–IV are protocol components in Figs. 1, 2) [19, 13]

Message 9 is integrity protected. Based on Message 9, the MS can verify that the SN received the CAP as the MS sent them. In addition, the integrity protecting as part of Message 9 allows the SN to authenticate itself to the MS.

Security in UMTS: The MS has correctly authenticated to the SN if *RES* matches *XRES*. The authentication of the SN to the MS requires two components to be present: the freshness of AUTN (sent in Message 7)—preventing the replay of authentication data—and a correctly integrity protected Message 9.

3.3 Roaming between GSM and UMTS

The UMTS standard defines roaming mechanisms to allow a GSM MS to connect to a UMTS infrastructure as well as for a UMTS MS to connect to a GSM infrastructure. The assumptions are that the respective MSs support both the GSM and UMTS radio interfaces and provide for both the GSM and UMTS encryption and integrity protection mechanisms.

For the discussion of roaming scenarios it is necessary to distinguish two components of the SN: the BS and the *Mobile Switching Center* (MSC). Depending on whether the BS and MSC are of GSM or UMTS technology, they are referred to as GSM/UMTS MSC and GSM/UMTS BS. There are six roaming scenarios:

1. A UMTS MS roams to a SN with a UMTS BS and UMTS MSC. This case is equivalent to regular UMTS operation.
2. A GSM MS roams to a SN with a GSM BS and GSM MSC. This case is equivalent to regular GSM operation.
3. A UMTS MS roams to a SN with a GSM BS and GSM MSC.
4. A GSM MS roams to a SN with a UMTS BS and UMTS MSC.
5. A UMTS MS roams to a SN with a GSM BS and a UMTS MSC. See Fig. 3.
6. A GSM MS roams to a SN with a GSM BS and a UMTS MSC. This case is equivalent to regular GSM operation. I.e., the UMTS MSC will act as a GSM MSC.

Security in Roaming: Scenarios (3) and (5) are prone to the false base station attack as the GSM cipher command is not integrity protected and does not include the MS's CAP. Furthermore, as shown in [13, 14] it is possible to mount an *asynchronous* man-in-the-middle attack in an all UMTS environment.

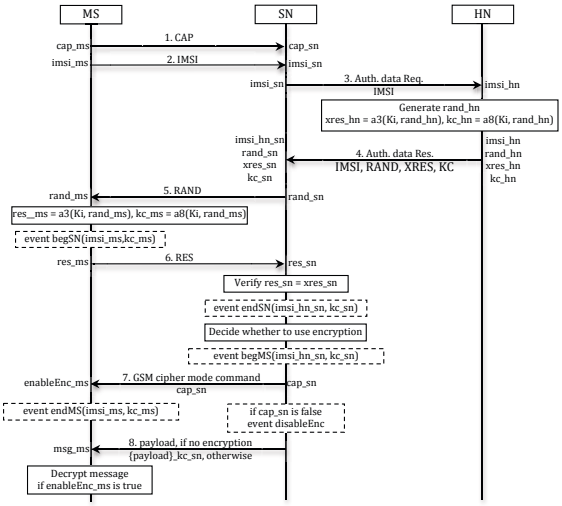

Fig. 4. GSM annotated in accord with our model, omitting TMSI/ID Request and adding a first message of data traffic

4 Modeling and Analyzing GSM and UMTS in ProVerif

4.1 GSM Model

Fig. 4 augments the protocol diagram of Fig. 1, as a guide to our model. The TMSI and ID request messages are omitted. When a MS first roams into a FN, an IMSI is always requested. Labels, like cap_ms and cap_sn for the first message, are the names of the relevant variables in the processes that model protocol roles. Besides variable names, the other augmentation is the addition of *events* (the dashed boxes). These are instrumentation used in order to specify authenticity properties as so-called *correspondence assertions*, which are the standard technique for formal specification of authenticity properties. Such properties take the form "if event E happens then event F must have happened previously". For example, whenever the SN decides to proceed with communication, using a particular K_c, because it successfully verified the response to a challenge, then that MS must indeed have sent the response and computed K_c for itself. An event is like a message—sent to an omniscient observer—with a tag indicating what kind of event, together with parameters like the specific IMSI and K_c. Events are not visible to the attacker, nor can they be generated by the attacker.

The HN, SN, and MS communicate over two shared channels declared in line 1 in Fig. 5. Traffic on the **private** channel is not accessible to the attacker; in our model, this channel is used by the SN and HN. Messages on a non-**private** channel can be copied, deleted, and fabricated by the attacker. We declare additional types and define a number of (distinct) constant values in lines 2–3. In lines 4–7, the cryptographic functions are declared and their algebraic properties are

```
1  free pubChannel: channel.  free secureChannel: channel [private].
2  type key. type ident. type nonce. type msgHdr. type resp. type sessKey.
3  const CAP: msgHdr. const ID: msgHdr. const AV_REQ: msgHdr. ...
4  fun a3(nonce, key) : resp. fun a8(nonce, key): sessKey.
5  fun sencrypt(bitstring, sessKey): bitstring.
6  reduc forall m: bitstring, k: sessKey;  sdecrypt(sencrypt(m, k), k) = m.
7  reduc encCapability() = true; encCapability() = false.
8  table keys(ident, key). (* Tables are not accessible to the attacker. *)
9  free payload: bitstring [private]. (*not initially known to attacker*)
10 free secretKc: bitstring [private].  (*a secret to be protected by Kc*)
```

Fig. 5. Excerpts from declarations for GSM

defined by equations. There are two equations in line 7 for encCapability so that the MS process can choose whether its capability includes encryption or not.

The main process forks multiple sessions for each kind of process. Fig. 6 shows the complete code of the processes. An instance of processMS models the initial activation of a MS followed by a single attempt at authentication. In lines 3–4, MS generates a fresh IMSI and a fresh K_i; in line 5 these are inserted in the table. Fresh values (keyword **new**) are unguessable by the attacker. In line 6, the MS decides on its encryption capability. The analysis will consider all possible executions, which thus include those where MS has encryption and those where it does not. In lines 8–9 the MS sends its capability and IMSI. Notice that the channel itself is not typed. The message in line 8 is the header literal CAP paired with boolean cap_ms; the one in line 9 pairs the ID header with a value of type ident. In line 10, the process waits until a message is available on the public channel. The message must match the designated format or the process terminates prematurely. PV statically checks types, which helps the user avoid modeling errors. However, PV ignores types during analysis. This enables PV to detect type flaw attacks. The events in lines 13 and 16 instrument the process to facilitate specification of security properties.

To be able to specify secrecy of data traffic following authentication, the model includes message *payload* declared in line 9 in Fig. 5. The SN encrypts payload, or not, based on the choice received in line 22 in Fig. 6. If encryption is not chosen, an event in line 34 records that fact. In line 41, HN uses the **get** operation on the private table of IMSI/K_i pairs for registered MSs.

Specifying and Analyzing Authenticity and Secrecy: We will consider in detail the following specifications:

```
1  query attacker(payload).
2  query attacker(payload) ⇝ event(disableEnc).
3  query attacker(secretKc).
4  query id: ident, k: sessKey;  event(endSN(id, k)) ⇝ event(begSN(id, k)).
5  query id: ident, k: sessKey;  event(endMS(id, k)) ⇝ event(begMS(id, k)).
```

Line 1 says that payload remains secret. This is not a requirement for GSM; the attacker can certainly obtain payload, e.g., if the MS is not capable of encryption. PV finds such attack trace. The conditional property in line 2 does express a requirement: if the attacker obtains the secret payload then the event disableEnc must have previously taken place in the SN. This query is indeed proved by PV.

```
1  let processMS =
2     (* registration and setup *)
3     new imsi_ms: ident;
4     new ki: key;
5     insert keys(imsi_ms, ki); (*pre-shared identity and key*)
6     let cap_ms: bool = encCapability() in (*choose capability*)
7     (* the protocol *)
8     out(pubChannel, (CAP, cap_ms));                    (*[Message 1]*)
9     out(pubChannel, (ID, imsi_ms));                    (*[Message 2]*)
10    in(pubChannel, (=CHALLENGE, rand_ms: nonce));      (*[Message 5]*)
11    let res_ms: resp = a3(rand_ms, ki) in (*compute response*)
12    let kc_ms: sessKey = a8(rand_ms, ki) in
13    event begSN(imsi_ms, kc_ms); (*MS is authenticating itself to SN*)
14    out(pubChannel, (RES, res_ms));                    (*[Message 6]*)
15    in(pubChannel, (=CMC, enableEnc_ms: bool));        (*[Message 7]*)
16    event endMS(imsi_ms, kc_ms);
17    in(pubChannel, (=MSG, msg_ms: bitstring));         (*[Message 8]*)
18    out(pubChannel, sencrypt(secretKc, kc_ms));
19    if enableEnc_ms = true then
20       let msgcontent: bitstring = sdecrypt(msg_ms, kc_ms) in  0.
21 let processSN =
22    in(pubChannel, (=CAP, cap_sn: bool));   (*[Message 1]*)
23    in(pubChannel, (=ID, imsi_sn: ident));  (*[Message 2]*)
24    out(secureChannel, (AV_REQ, imsi_sn));  (*[Message 3]*)
25    in(secureChannel,                       (*[Message 4]*)
26     (=AV, imsi_hn_sn:ident, rand_sn:nonce, xres_sn:resp, kc_sn:sessKey));
27    out(pubChannel, (CHALLENGE, rand_sn));  (*[Message 5]*)
28    in(pubChannel, (=RES, res_sn: resp));   (*[Message 6]*)
29    if res_sn = xres_sn then    (*Check response*)
30       event endSN(imsi_hn_sn, kc_sn);
31       event begMS(imsi_hn_sn, kc_sn);
32       out(pubChannel, (CMC, cap_sn));      (*[Message 7]*)
33       if cap_sn = false then
34          event disableEnc;
35          out(pubChannel, (MSG, payload))   (*[Message 8]*)
36       else
37          out(pubChannel, (MSG, sencrypt(payload, kc_sn))). (*[Message 8]*)
38 let processHN =
39     in(secureChannel, (=AV_REQ, imsi_hn: ident)); (*[Message 3]*)
40     new rand_hn: nonce; (*Generate a fresh random number*)
41     get keys(=imsi_hn, ki_hn) in  (*attempt to look up key*)
42     let xres_hn: resp = a3(rand_hn, ki_hn) in (*compute XRES and Kc*)
43     let kc_hn: sessKey = a8(rand_hn, ki_hn) in
44     out(secureChannel, (AV, imsi_hn, rand_hn, xres_hn, kc_hn)). (*[Message 4]*)
```

Fig. 6. Processes for GSM, with reference to Fig. 4

Another requirement is that K_i and K_c remain secret—regardless of whether encryption is enabled. We declare another secret in line 10 in Fig. 5 and specify that the attacker does not obtain it. Then we introduce an extra message in line 18 of the MS process, and it is sent regardless of whether encryption is enabled. The query in line 3 is successful, which implies that K_c remains secret since otherwise the attacker would obtain secretKc. The property is unconditional: the key remains secret regardless of whether encryption is chosen. Furthermore, because K_c is computed as a function of $RAND$, which is sent in the clear, and K_i, secrecy of K_c implies secrecy of K_i.

Line 4 says that whenever the event endSN occurs with some arguments id and k, there must have been a prior occurrence of event begSN with the same arguments. The former event happens only following successful check of the expected

response from MS. This query says that if the SN believes it has established a session key K_c associated with an MS using this particular IMSI, for which the HN has provided a challenge and expected response, then indeed there is a MS that reached that stage of its protocol role, for that IMSI and K_c. The event endSN occurs only following successful verification in the SN, as that is the step that is intended to establish authentication. The event begSN is placed in the MS process just before it sends the response, i.e., after it has computed the K_c to which the event refers. The event should not be placed later in the MS process, because successful verification by the SN does not give the SN evidence that MS has progressed any further.

The protocol is not intended to authenticate the SN to MS (see Sect. 3.1). However, to gain confidence in our model we check that property. For the query in line 5 above, PV does find a trace that violates the property.

A more interesting man-in-the-middle scenario is the one PV finds in violation of the first secrecy condition above, **query attacker**(payload). The MS sends out its encryption capability and identification. The attacker intercepts the capability message and changes it to no-encryption, so the SN receives the modified capability. The HN generates the authentication vector and sends it to the SN. The one-way challenge-response between the SN and MS succeeds. Since the SN receives no-encryption from the attacker, it decides not to use encryption.

4.2 Modeling and Analyzing UMTS

As with GSM, we omit the TMSI and ID Request messages. We also abstract from the SQN and AMF. UMTS authentication establishes the *CK* and the *IK*, so these are included in the parameters of the events. The secrecy and authentication properties are specified similarly as in the GSM model. The simple secrecy property, **attacker**(payload), fails: as in GSM, the MS can choose no-encryption, and regardless of the choice the attacker can modify the capability message to claim the MS has no encryption. However, if the MS chooses the capability of encryption and the attacker modifies that, this will be detected by the MS which will stop responding. PV proves the conditional secrecy property **attacker**(payload)⇝**event**(disableEnc) as well as secrecy of *CK* and *IK* (using the idiom explained in Sect. 4.1). It also proves authentication in both directions:

```
query d:ident,c:cipherKey,i:integKey;event(endSN(d,c,i))⇝event(begSN(d,c,i)).
query d:ident,c:cipherKey,i:integKey;event(endMS(d,c,i))⇝event(begMS(d,c,i)).
```

5 Modeling and Analyzing GSM/UMTS Roaming

As described in Sect. 3.3, there are six different roaming scenarios, three of which are basically the same as GSM or UMTS from the perspective of security. Of the other three cases, numbers (3–5), case (5) is the most interesting.

In case (5), to support GSM BS, the SN converts the UMTS keys into a GSM session key. The CMC message does not include the received encryption capability and is not integrity protected. To communicate with GSM BS, the

MS converts the UMTS keys into a GSM session key. Then the MS performs the steps in GSM block IV. The HN is modeled exactly the same as in UMTS.

Since the SN authenticates the MS as in UMTS authentication, the events begMS and endMS use UMTS keys in the authentication property specification:

```
1  query attacker(payload).
2  query attacker(payload) ⇝ event(disableEnc).
3  query d:ident, c:cipherKey, i:integKey;
4        event(endSN(d,c,i)) ⇝ event(begSN(d,c,i)).
5  query d:ident, k:sessKey; event(endMS(d,k)) ⇝ event(begMS(d,k)).
```

The query of secrecy of payload in line 1 fails and PV finds a trace similar to the one we describe for the same query against GSM. The required properties in lines 2–4 are proved, as is key secrecy.

What is interesting is that the required authentication property in line 5 is violated. In the attack trace, the attacker first acts as a BS to intercept the CAP message and replace it with no-encryption. The attacker forwards the challenge and response. The MS receives the CMC message which is forged by the attacker.

References

1. Abadi, M., Blanchet, B., Fournet, C.: Just Fast Keying in the Pi Calculus. In: Schmidt, D. (ed.) ESOP 2004. LNCS, vol. 2986, pp. 340–354. Springer, Heidelberg (2004)
2. Blanchet, B., Chaudhuri, A.: Automated formal analysis of a protocol for secure file sharing on untrusted storage. In: IEEE Symp. on Sec. and Priv. (2008)
3. Bodei, C., Buchholtz, M., Degano, P., Nielson, F., Nielson, H.R.: Automatic validation of protocol narration. In: IEEE CSFW, pp. 126–140 (2003)
4. Bouassida, M.S., Chridi, N., Chrisment, I., Festor, O., Vigneron, L.: Automated verification of a key management architecture for hierarchical group protocols. Annals of Telecommunications (2007)
5. Chang, R., Shmatikov, V.: Formal analysis of authentication in Bluetooth device pairing. In: FCS-ARSPA (2007)
6. Dunkelman, O., Keller, N., Shamir, A.: A Practical-Time Related-Key Attack on the KASUMI Cryptosystem Used in GSM and 3G Telephony. In: Rabin, T. (ed.) CRYPTO 2010. LNCS, vol. 6223, pp. 393–410. Springer, Heidelberg (2010)
7. Escobar, S., Meadows, C., Meseguer, J.: State Space Reduction in the Maude-NRL Protocol Analyzer. In: Jajodia, S., Lopez, J. (eds.) ESORICS 2008. LNCS, vol. 5283, pp. 548–562. Springer, Heidelberg (2008)
8. Fox, D.: Der IMSI catcher. In: DuD Datenschutz und Datensicherheit (2002)
9. Golić, J.D.: Cryptanalysis of Alleged A5 Stream Cipher. In: Fumy, W. (ed.) EUROCRYPT 1997. LNCS, vol. 1233, pp. 239–255. Springer, Heidelberg (1997)
10. Jakobsson, M., Wetzel, S.: Security Weaknesses in Bluetooth. In: Naccache, D. (ed.) CT-RSA 2001. LNCS, vol. 2020, pp. 176–191. Springer, Heidelberg (2001)
11. Mitchell, C.J., Knudsen, L.R.: An analysis of the 3GPP-MAC scheme. In: WCC (2001)
12. Lim, S.-H., Bang, K.-S., Yi, O., Lim, J.: A Secure Handover Protocol Design in Wireless Networks with Formal Verification. In: Boavida, F., Monteiro, E., Mascolo, S., Koucheryavy, Y. (eds.) WWIC 2007. LNCS, vol. 4517, pp. 67–78. Springer, Heidelberg (2007)

13. Meyer, U.: Secure Roaming and Handover Procedures in Wireless Access Networks. PhD thesis, Darmstadt University of Technology, Germany (2005)
14. Meyer, U., Wetzel, S.: A man-in-the-middle attack on UMTS. In: ACM WiSec, pp. 90–97 (2004)
15. Meyer, U., Wetzel, S.: On the impact of GSM encryption and man-in-the-middle attacks on the security of interoperating GSM/UMTS networks. In: IEEE Symposium on Personal, Indoor and Mobile Radio Communications (2004)
16. Niemi, V., Nynberg, K.: UMTS Security. Wiley (2003)
17. Taha, A., Abdel-Hamid, A., Tahar, S.: Formal analysis of the handover schemes in mobile WiMAX networks. In: Conf. on Wireless and Optical Comm. Net. (2009)
18. Taha, A., Abdel-Hamid, A., Tahar, S.: Formal verification of IEEE 802.16 security sublayer using Scyther tool. In: IFIP N2S 2009, pp. 1–6 (2009)
19. 3GPP The mobile broadband standard, `http://www.3gpp.org/specifications`