Formal verification of the 5G EAP-TLS authentication protocol using Proverif

2023

Alessandro Zanatta

University of Pisa

Reference paper (DOI): 10.1109/ACCESS.2020.2969474

Proverif

- Symbolic verification tool:

Proverif

- Symbolic verification tool:
  - Attacker $\longrightarrow$ Dolev-Yao;

Proverif

- Symbolic verification tool:
  - Attacker $\longrightarrow$ Dolev-Yao;
  - Messages $\longrightarrow$ terms;

Proverif

- Symbolic verification tool:
  - Attacker $\longrightarrow$ Dolev-Yao;
  - Messages $\longrightarrow$ terms;
  - Cryptographic primitives $\longrightarrow$ black-box;

Proverif

- Symbolic verification tool:
    - Attacker $\longrightarrow$ Dolev-Yao;
    - Messages $\longrightarrow$ terms;
    - Cryptographic primitives $\longrightarrow$ black-box;
    - Perfect cryptography assumption.

Proverif

- Symbolic verification tool:
  - Attacker $\longrightarrow$ Dolev-Yao;
  - Messages $\longrightarrow$ terms;
  - Cryptographic primitives $\longrightarrow$ black-box;
  - Perfect cryptography assumption. Suppose we have:
    - Two primitives: enc, dec;
    - Two terms: m, k;
    - The following equality:

$$\mathrm{dec}\left(\mathrm{enc}\left(m, k\right), k\right) = m \tag{1}$$

Proverif

- Symbolic verification tool:
  - Attacker $\longrightarrow$ Dolev-Yao;
  - Messages $\longrightarrow$ terms;
  - Cryptographic primitives $\longrightarrow$ black-box;
  - Perfect cryptography assumption. Suppose we have:
    - Two primitives: enc, dec;
    - Two terms: m, k;
    - The following equality:

$$\mathrm{dec}\left(\mathrm{enc}\left(m, k\right), k\right) = m \tag{1}$$

    - can decrypt $\mathrm{enc}\left(m, k\right) \iff k$ is known

Proverif

- Symbolic verification tool:
  - Attacker $\longrightarrow$ Dolev-Yao;
  - Messages $\longrightarrow$ terms;
  - Cryptographic primitives $\longrightarrow$ black-box;
  - Perfect cryptography assumption. Suppose we have:
    - Two primitives: enc, dec;
    - Two terms: m, k;
    - The following equality:

$$\mathrm{dec}\left(\mathrm{enc}\left(m, k\right), k\right) = m \tag{1}$$

    - can decrypt $\mathrm{enc}\left(m, k\right) \iff k$ is known
- Based on applied π-calculus;

Applied π-calculus

Grammar of processes (P, Q):

Applied π-calculus

Grammar of processes (P, Q):

```
0                    (* null process *)
```

Applied π-calculus

Grammar of processes (P, Q):

```
0                       (* null process *)
out(N, M); P            (* output to channel N the message M *)
in(N, M: T); P          (* input from channel N of message M *)
```

Applied π-calculus

Grammar of processes (P, Q):

```
0                       (* null process *)
out(N, M); P            (* output to channel N the message M *)
in(N, M: T); P          (* input from channel N of message M *)
P | Q                   (* parallel composition *)
!P                      (* infinite replication *)
```

Applied π-calculus

Grammar of processes (P, Q):

```
0                       (* null process *)
out(N, M); P            (* output to channel N the message M *)
in(N, M: T); P          (* input from channel N of message M *)
P | Q                   (* parallel composition *)
!P                      (* infinite replication *)
new a: T; P             (* fresh value of sort T *)
if M then P else Q      (* conditional *)
```

Applied π-calculus

Grammar of processes (P, Q):

```
0                     (* null process *)
out(N, M); P          (* output to channel N the message M *)
in(N, M: T); P        (* input from channel N of message M *)
P | Q                 (* parallel composition *)
!P                    (* infinite replication *)
new a: T; P           (* fresh value of sort T *)
if M then P else Q    (* conditional *)
```

Additionally:

```
event EventName(x);         (* add event to trace *)
query event(EventName(x)). (* define a query on events *)
```

Applied π-calculus

Grammar of processes (P, Q):

```
0                      (* null process *)
out(N, M); P           (* output to channel N the message M *)
in(N, M: T); P         (* input from channel N of message M *)
P | Q                  (* parallel composition *)
!P                     (* infinite replication *)
new a: T; P            (* fresh value of sort T *)
if M then P else Q     (* conditional *)
```

Additionally:

```
event EventName(x);           (* add event to trace *)
query event(EventName(x)).    (* define a query on events *)
let macroName = P.            (* create a process macro *)
let x = M in P else Q.        (* assignment and pattern matching *)
```

Applied π-calculus

Grammar of processes (P, Q):

```
0                     (* null process *)
out(N, M); P          (* output to channel N the message M *)
in(N, M: T); P        (* input from channel N of message M *)
P | Q                 (* parallel composition *)
!P                    (* infinite replication *)
new a: T; P           (* fresh value of sort T *)
if M then P else Q    (* conditional *)
```

Additionally:

```
event EventName(x);         (* add event to trace *)
query event(EventName(x)).  (* define a query on events *)
let macroName = P.          (* create a process macro *)
let x = M in P else Q.      (* assignment and pattern matching *)
phase t;                    (* execute a process in phase t *)
```

5G EAP-TLS protocol entities

Involved entities:

- User Equipment (UE):
  - Subscription Permanent Identifier (SUPI)
  - Public asymmetric key pk_{HN}

5G EAP-TLS protocol entities

Involved entities:

- User Equipment (UE):
  - Subscription Permanent Identifier (SUPI)
  - Public asymmetric key pk_{HN}
- Home Network (HN):
  - Authentication Server Function (AUSF)
  - Unified Data Management (UDM)

5G EAP-TLS protocol entities

Involved entities:

- User Equipment (UE):
  - Subscription Permanent Identifier (SUPI)
  - Public asymmetric key pk_{HN}
- Home Network (HN):
  - Authentication Server Function (AUSF)
  - Unified Data Management (UDM)
- Serving Network (SN):
  - Security Anchor Function (SEAF)

5G EAP-TLS protocol entities

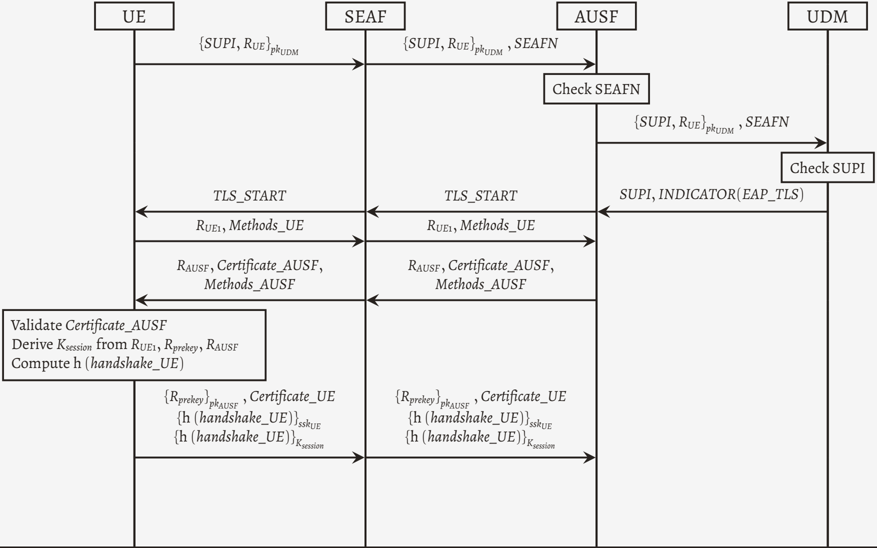
Involved entities:

- User Equipment (UE):
  - Subscription Permanent Identifier (SUPI)
  - Public asymmetric key pk_{HN}
- Home Network (HN):
  - Authentication Server Function (AUSF)
  - Unified Data Management (UDM)
- Serving Network (SN):
  - Security Anchor Function (SEAF)

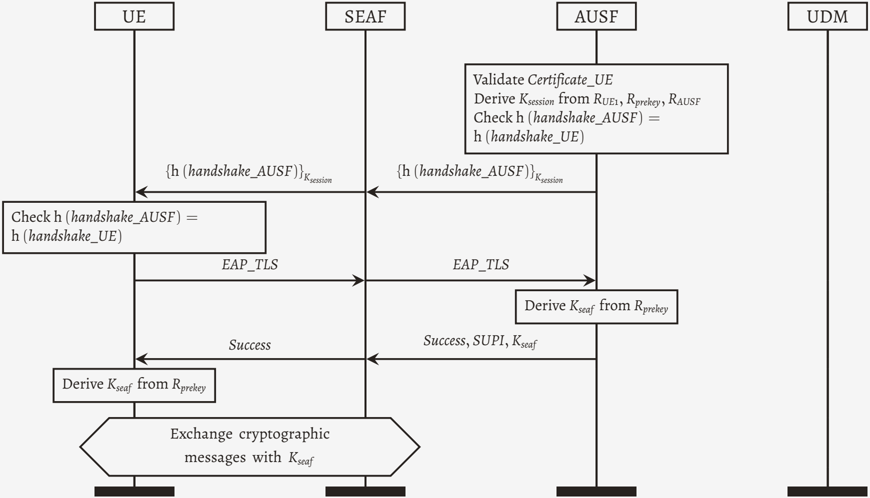
Assumptions:

- HN $\leftrightarrow$ SN communications are secure

5G EAP-TLS protocol execution I



5G EAP-TLS protocol execution II



Required security properties

Required security properties

- **Authentication properties**:
  - A1. Both the home network and the subscriber should agree on the identity of each other after successful termination
  - A2. Both the home network and the subscriber should agree on the pre-master key R_{prekey} after successful termination

Required security properties

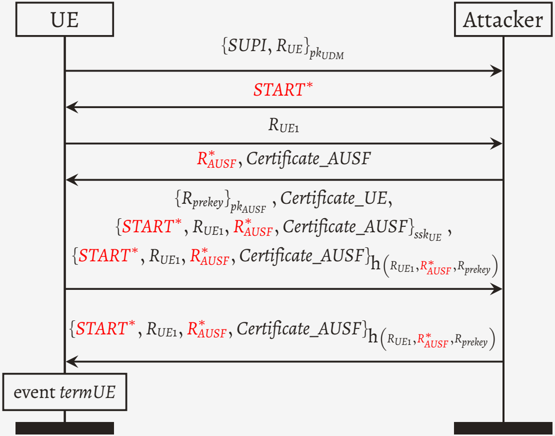
- **Authentication properties**:
  - A1. Both the home network and the subscriber should agree on the identity of each other after successful termination
  - A2. Both the home network and the subscriber should agree on the pre-master key R_{prekey} after successful termination
- **Secrecy properties**:
  - S1. The attacker cannot obtain the identity *SUPI* of an honest subscriber
  - S2. The attacker cannot obtain the pre-master key R_{prekey} of an honest subscriber
  - S3. The attacker cannot obtain the session key $K_{session}$ of an honest subscriber

It's ***DEMO*** time!!

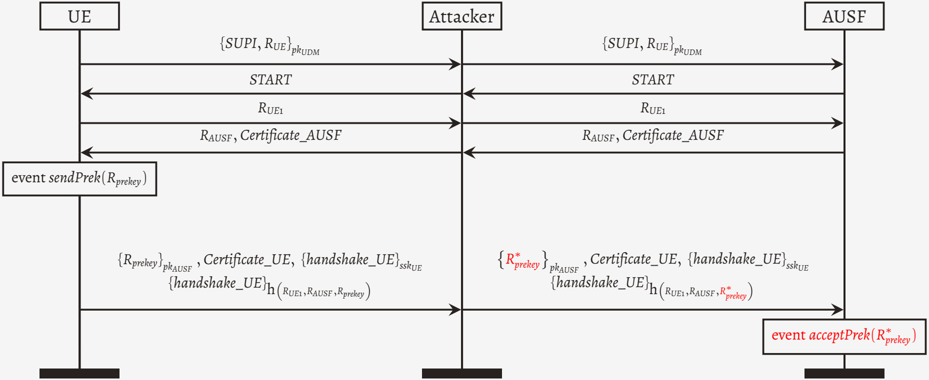
Broken properties

- **Authentication properties**:
  A1. Both the home network and the subscriber should agree on the identity of each other after successful termination
  A2. Both the home network and the subscriber should agree on the pre-master key R_{prekey} after successful termination
- **Secrecy properties**:
  S1. The attacker cannot obtain the identity *SUPI* of an honest subscriber
  S2. The attacker cannot obtain the pre-master key R_{prekey} of an honest subscriber
  S3. The attacker cannot obtain the session key $K_{session}$ of an honest subscriber

Counterexample for property A1



Counterexample for property A2



Fixing the protocol