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 → Dolev-Yao;

Proverif

- Symbolic verification tool:
  - Attacker ⟶ Dolev-Yao;
  - Messages ⟶ terms;

Proverif

- Symbolic verification tool:
  - Attacker → Dolev-Yao;
  - Messages → terms;
  - Cryptographic primitives → black-box;

Proverif

- Symbolic verification tool:
    - Attacker → Dolev-Yao;
    - Messages → terms;
    - Cryptographic primitives → 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:

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

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:

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

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

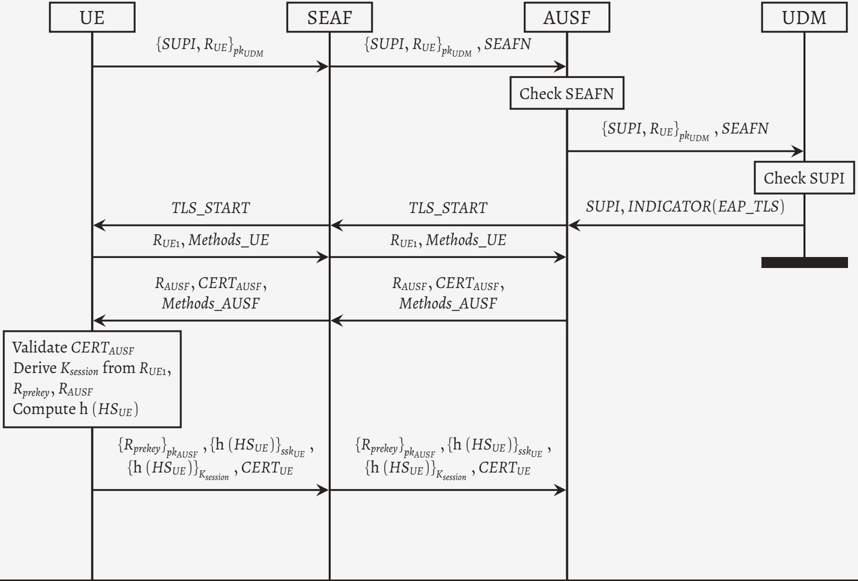
Involved entities:

- 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



UE

SEAF

AUSF

UDM

$\{SUPI, R_{UE}\}_{pk_{UDM}}$

$\{SUPI, R_{UE}\}_{pk_{UDM}}, SEAFN$

Check SEAFN

$\{SUPI, R_{UE}\}_{pk_{UDM}}, SEAFN$

Check SUPI

$SUPI, INDICATOR(EAP\_TLS)$

TLS_START

TLS_START

$R_{UE1}, Methods\_UE$

$R_{UE1}, Methods\_UE$

$R_{AUSF}, CERT_{AUSF}, Methods\_AUSF$

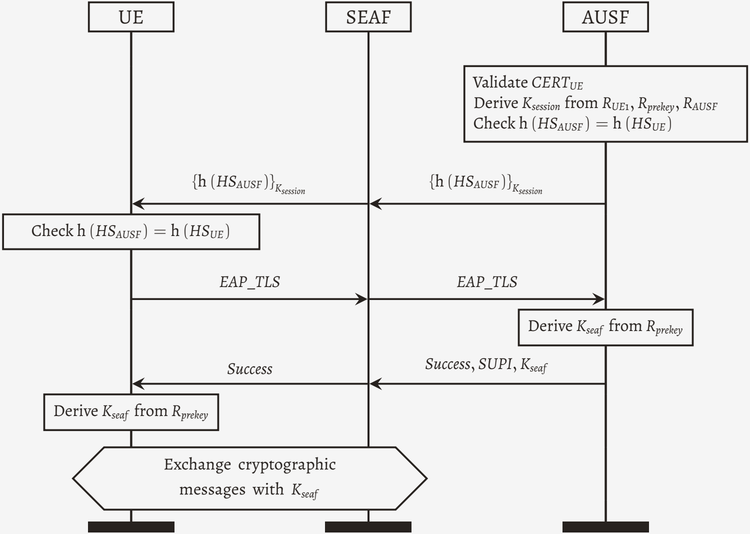
$R_{AUSF}, CERT_{AUSF}, Methods\_AUSF$

Validate $CERT_{AUSF}$
Derive $K_{session}$ from R_{UE1}, R_{prekey}, R_{AUSF}
Compute h (HS_{UE})

$\{R_{prekey}\}_{pk_{AUSF}}, \{\text{h}(HS_{UE})\}_{ssk_{UE}}, \{\text{h}(HS_{UE})\}_{K_{session}}, CERT_{UE}$

$\{R_{prekey}\}_{pk_{AUSF}}, \{\text{h}(HS_{UE})\}_{ssk_{UE}}, \{\text{h}(HS_{UE})\}_{K_{session}}, CERT_{UE}$

5G EAP-TLS protocol execution II


UE
SEAF
AUSF
Validate $CERT_{UE}$
Derive $K_{session}$ from $R_{UE1}, R_{prekey}, R_{AUSF}$
Check $\mathrm{h}(HS_{AUSF}) = \mathrm{h}(HS_{UE})$
$\{\mathrm{h}(HS_{AUSF})\}_{K_{session}}$
$\{\mathrm{h}(HS_{AUSF})\}_{K_{session}}$
Check $\mathrm{h}(HS_{AUSF}) = \mathrm{h}(HS_{UE})$
EAP_TLS
EAP_TLS
Derive K_{seaf} from R_{prekey}
$Success$
$Success, SUPI, K_{seaf}$
Derive K_{seaf} from R_{prekey}
Exchange cryptographic messages with K_{seaf}

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

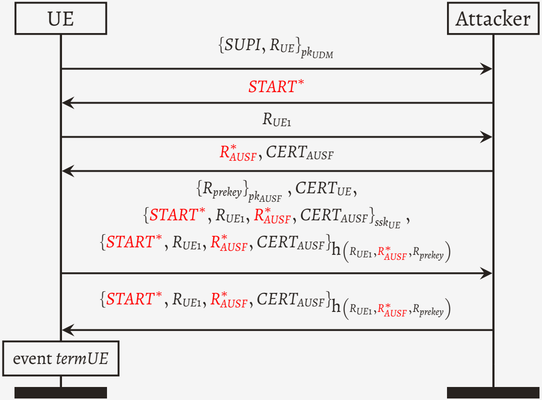
- **Authentication properties**:
  - A1. Both the home network and the subscriber should agree on the identity of each other after successful termination
  - A2. Both the home network and the subscriber should agree on the pre-master key R_{prekey} after successful termination
- **Secrecy properties**: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



UE

Attacker

$\{SUPI, R_{UE}\}_{pk_{UDM}}$

$START^*$

R_{UE1}

$R^*_{AUSF}, CERT_{AUSF}$

$\{R_{prekey}\}_{pk_{AUSF}}, CERT_{UE},$

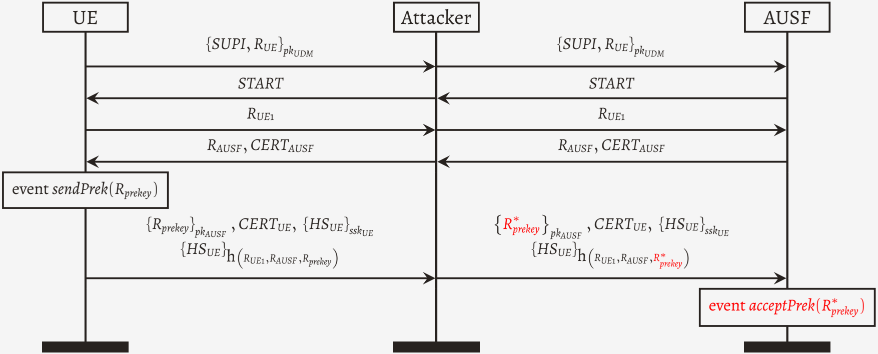
$\{START^*, R_{UE1}, R^*_{AUSF}, CERT_{AUSF}\}_{ssk_{UE}},$

$\{START^*, R_{UE1}, R^*_{AUSF}, CERT_{AUSF}\}_{h(R_{UE1}, R^*_{AUSF}, R_{prekey})}$

$\{START^*, R_{UE1}, R^*_{AUSF}, CERT_{AUSF}\}_{h(R_{UE1}, R^*_{AUSF}, R_{prekey})}$

event *termUE*

Counterexample for property A2



UE

Attacker

AUSF

$\{SUPI, R_{UE}\}_{pk_{UDM}}$

$\{SUPI, R_{UE}\}_{pk_{UDM}}$

$START$

$START$

R_{UE1}

R_{UE1}

$R_{AUSF}, CERT_{AUSF}$

$R_{AUSF}, CERT_{AUSF}$

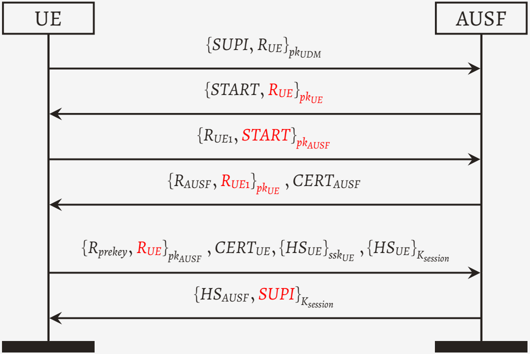
event $sendPrek(R_{prekey})$

$\{R_{prekey}\}_{pk_{AUSF}}, CERT_{UE}, \{HS_{UE}\}_{ssk_{UE}}$
$\{HS_{UE}\}_{\mathrm{h}(R_{UE1}, R_{AUSF}, R_{prekey})}$

$\{R^{*}_{prekey}\}_{pk_{AUSF}}, CERT_{UE}, \{HS_{UE}\}_{ssk_{UE}}$
$\{HS_{UE}\}_{\mathrm{h}(R_{UE1}, R_{AUSF}, R^{*}_{prekey})}$

event $acceptPrek(R^{*}_{prekey})$

Fixing the protocol



UE

AUSF

$\{SUPI, R_{UE}\}_{pk_{UDM}}$

$\{START, R_{UE}\}_{pk_{UE}}$

$\{R_{UE1}, START\}_{pk_{AUSF}}$

$\{R_{AUSF}, R_{UE1}\}_{pk_{UE}}, CERT_{AUSF}$

$\{R_{prekey}, R_{UE}\}_{pk_{AUSF}}, CERT_{UE}, \{HS_{UE}\}_{ssk_{UE}}, \{HS_{UE}\}_{K_{session}}$

$\{HS_{AUSF}, SUPI\}_{K_{session}}$