Towards nonce-misuse resistant protocols

Tristan Claverie*, joint work with Gildas Avoine, Stéphanie Delaune

*: Agence nationale de la sécurité des systèmes d’information

Apr 05, 2024
Incorrect nonce handling: protocol implementations

- WPS\(^1\): predictable nonces $\Rightarrow$ authentication bypass, session key leaks
- LoRaWAN 1.0\(^2\): reused nonces $\Rightarrow$ replay, messages confidentiality weakens
- Bluetooth secure numeric comparison\(^3\): predictable nonces $\Rightarrow$ authentication bypass, session key leaks

\[\begin{align*}
1. & \quad \textit{Offline bruteforce attack on WiFi Protected Setup}, \ D. \ Bongard, \ Hack.Lu \ '14 \\
2. & \quad \textit{Rescuing LoRaWAN 1.0}, \ G. \ Avoine \ and \ L. \ Ferreira, \ FC \ '18 \\
3. & \quad \textit{Bluetooth Randomness is Mostly Random}, \ J. \ Tillmanns, \ J. \ Classen, \ F. \ Rohrbach, \ and \ M.\Hollick, \\
& \quad \textit{WOOT} \ '20
\end{align*}\]
- Misuse-resistance defined in 2006 for authenticated encryption\(^4\)
- Some authenticated encryption modes have been designed to be misuse-resistant

---

4. A Provable-Security Treatment of the Key-Wrap Problem, P. Rogaway and T. Shrimpton, EUROCRYPT '06
Nonces are perfect?
Using symbolic models:

- Incorrect implementation of nonces: what impact?
- Are there more protocols more resilient than others?
1. Tackling nonce misuse in symbolic models

2. A proposed Tamarin representation

3. Implementation and results

4. Conclusion
1. Tackling nonce misuse in symbolic models
Nonce properties

- Freshness
- Unpredictability
Modelling nonce predictability

Towards nonce-misuse resistant protocols

T.Claverie (ANSSI)
Modelling nonce predictability

Towards nonce-misuse resistant protocols
Which nonce is considered to be reused? when?

Example protocol:

```
A   B
N_{a1}  
N_{b1}  
N_{a2}  
N_{b2}  
```
Which nonce is considered to be reused? when?

Example protocol:

\[ \begin{array}{c}
A \\
\uparrow \quad \downarrow \\
N_{a1} \\
\downarrow \quad \uparrow \\
N_{b1} \\
\downarrow \quad \uparrow \\
N_{a2} \\
\downarrow \quad \uparrow \\
N_{b2} \\
B \\
\end{array} \]
Which nonce is considered to be reused? When?

Example protocol:

Deriving nonce reuse cases:

Towards nonce-misuse resistant protocols

T.Claverie (ANSSI)
Which nonce is considered to be reused? when?

Example protocol:

Deriving nonce reuse cases:
Which nonce is considered to be reused? when?

Example protocol:

```
A    B
Na1  Nb1
Na2  Nb2
```

Deriving nonce reuse cases:

```
A    B
Na1  Na1
Na2  Na2
```

Towards nonce-misuse resistant protocols
Which nonce is considered to be reused? when?

Example protocol:

A

\[ N_{a1} \]

\[ N_{b1} \]

\[ N_{a2} \]

\[ N_{b2} \]

B

Deriving nonce reuse cases:

A

\[ N_{a1} \]

\[ N_{b1} \]

\[ N_{a2} \]

\[ N_{b2} \]

B

A

\[ N_{a1} \]

\[ N_{b1} \]

\[ N_{a2} \]

\[ N_{b2} \]

B
Considering a protocol between two agents, two roles, for an unbounded number of sessions:

- Same agent, same role, same session;
- Same agent, same role, different session;
- Same agent, different role, different session;
- Different agent, different role, same session;
- ...
2. A proposed Tamarin representation
Idea : (Mis)generate nonces in a specific rule
Idea: (Mis)generate nonces in a specific rule

```
Fr(∼n)
A(∼n)
A(∼n)
Fr(∼n)
Nonce(∼n)
OnlyOnce()
Nonce(∼n),Nonce(∼n)
```
Idea: (Mis)generate nonces in a specific rule

\[
\begin{align*}
& \text{Fr}(\sim n) \\
& \text{Nonce}(\sim n) \\
& \text{Nonce}(\sim n) \\
& \text{A}(\sim n) \\
& \text{A}(\sim n) \\
& \text{B}(\sim n)
\end{align*}
\]
Idea: (Mis)generate nonces in a specific rule

\[
\begin{array}{c}
\text{Fr}(\sim n) \\
\text{Nonce}(\sim n) \\
\text{Nonce}(\sim n) \\
\text{A}(\sim n) \\
\text{A}(\sim n) \\
\text{B}(\sim n)
\end{array}
\]

\[
\begin{array}{c}
\text{Fr}(\sim n) \\
\text{OnlyOnce()} \\
\text{Nonce}(\sim n), \text{Nonce}(\sim n)
\end{array}
\]
Option 1: \texttt{Nonce(\sim n)}

- The same nonce may be used twice for different agents/roles/sessions
  
  - If security properties are proven, any kind of nonce reuse has no impact
  
  - If they are not, which scenarios hold and which do not?
Option 1: $\text{Nonce}(\sim n)$

- The same nonce may be used twice for different agents/roles/sessions
- $\rightarrow$ If security properties are proven, any kind of nonce reuse has no impact
- $\rightarrow$ If they are not, which scenarios hold and which do not?

Option 2 (outline): $\text{Nonce}(\text{\textquoteleft id\textquoteright}, \sim n)$

- Add identifiers to $\text{Nonce}(\ldots)$ facts
- Those identifier allow to misgenerate nonces only for a single agent/role/session
Modelling reuse:

- Multiple option possibles, some alternatives may exist
- Modelling choice depends on what is studied:
  - Option 1: Easy to modify 'Fr' with 'Nonce', hard to understand which scenario fails
  - Option 2: More difficult to modify the protocol, easy to understand the case studied
3. Implementation and results
Generating misuse cases

- We introduce the special Tamarin fact "Nonce"
- Implement a Tamarin parser to get the Abstract Syntax Tree
- Modify the AST to generate all possible misuse cases
- Output models are analysed separately with Tamarin.
Case study: LoRaWAN 1.0

Device

new DevNonce

$m1 = \langle \text{AppID, DevID, DevNonce} \rangle$

$mac(\text{AppKey, } m1)$

Network

new JoinNonce

$enc(\text{AppKey, } \langle \text{DevAddr, JoinNonce, }\rangle$

$mac(\text{AppKey, } \langle \text{DevAddr, JoinNonce} \rangle)$

$AppSKey = enc(\text{AppKey, } \langle '1', \text{JoinNonce, DevNonce} \rangle)$

$NwkSKey = enc(\text{AppKey, } \langle '2', \text{JoinNonce, DevNonce} \rangle)$
Generated cases:

- Basic case: nonces are fresh and unpredictable
- 2 leak cases: DevNonce and JoinNonce leak as soon as they are generated
- 4 reuse cases:
  - Device reuses DevNonce in two sessions
  - Network reuses JoinNonce in two sessions
  - Device always uses the same DevNonce
  - Network always uses the same JoinNonce
<table>
<thead>
<tr>
<th>Case</th>
<th>Agr D</th>
<th>Agr N</th>
<th>Inj Agr D</th>
<th>Inj Agr N</th>
<th>Fresh Session</th>
<th>Key N</th>
<th>Fresh Session</th>
<th>Key D</th>
</tr>
</thead>
<tbody>
<tr>
<td>Basic</td>
<td>✓</td>
<td>✓</td>
<td>✗</td>
<td>✗</td>
<td>✓</td>
<td></td>
<td></td>
<td>✓</td>
</tr>
<tr>
<td>Leak D</td>
<td>✓</td>
<td>✓</td>
<td>✗</td>
<td>✗</td>
<td>✓</td>
<td></td>
<td></td>
<td>✓</td>
</tr>
<tr>
<td>Leak N</td>
<td>✓</td>
<td>✓</td>
<td>✗</td>
<td>✗</td>
<td>✓</td>
<td></td>
<td></td>
<td>✓</td>
</tr>
<tr>
<td>Reuse Once D</td>
<td>✓</td>
<td>✓</td>
<td>✗</td>
<td>✗</td>
<td>✗</td>
<td></td>
<td></td>
<td>✗</td>
</tr>
<tr>
<td>Reuse Once N</td>
<td>✓</td>
<td>✓</td>
<td>✗</td>
<td>✗</td>
<td>✗</td>
<td></td>
<td></td>
<td>✗</td>
</tr>
<tr>
<td>Reuse Always D</td>
<td>✓</td>
<td>✓</td>
<td>✗</td>
<td>✗</td>
<td>✗</td>
<td></td>
<td></td>
<td>✗</td>
</tr>
<tr>
<td>Reuse Always N</td>
<td>✓</td>
<td>✓</td>
<td>✗</td>
<td>✗</td>
<td>✗</td>
<td></td>
<td></td>
<td>✗</td>
</tr>
</tbody>
</table>
ISO 9798 key agreement protocols:

- 9798-2-1: injective agreement fails on both sides when nonce reused
- 9798-2-4: injective agreement fails on both sides when nonce reused
- 9798-2-6: injective agreement fails on one side when nonce reused
- 9798-3-4: injective agreement fails on both sides when nonce reused
- 9798-4-4: injective agreement fails on both sides when nonce reused
Towards nonce-misuse resistant protocols

T.Claverie (ANSSI) Apr 05, 2024 18 / 20
4. Conclusion
Identify two core properties for nonces, that are not always granted by implementations

Define ways to model misimplementation of those properties in Tamarin

Study some concrete protocols with the modifications made
Future work

- More complex case studies
- Identify additional possible misuses
- Find resistant protocols?