Blind signature as a shield against backdoors in smart-cards

Liliya Akhmetzyanova, Evgeny Alekseev, Alexandra Babueva, Andrey Bozhko, Stanislav Smyshlyaev
Outline

1. Motivation
2. Related work
3. Blind signatures
4. Our contribution
5. Open problems
1. Motivation
2. Related work
3. Blind signatures
4. Our contribution
5. Open problems
The uploaded user signing key is protected against adversary-thief which can get physical access to smart-card:

- Engineering protection against physical key extraction;
- Password-based protected access to signing API (e.g. PAKE).
The signing code is often hardwired into smart-card microchips and cannot be openly verified. This makes it possible for unscrupulous developers to implement a malicious code.
Example: in case of using the GOST signature scheme, malicious smart-card can sample low-entropy one-time values $k$ allowing the developer to recover the user key from one correct signature.
Outline

1. Motivation
2. Related work
3. Blind signatures
4. Our contribution
5. Open problems
Previously on …


It proposes:
1. to consider two type of adversaries:
   • External (to smart-card) adversary
   • Adversary with agent (malicious smart-card)
2. solution for the GOST signature scheme to protect against such adversaries.
External adversaries

Smart-card with private key

trusted application with memory leak

message

signature

(message, signature)

Internet

Honest-but-curious adversary acting on the application side (computer virus).

Goal: to make a forgery, in particular, by key recovery
Adversary with agent

Adversary consists of two parts:
1. an active adversary on the smart-card side (smart-card with backdoor).
2. a passive adversary collecting correct signatures (backdoor implementer).

Goal: to make a forgery, in particular, by key recovery
Idea of the solution: additional usage of Schnorr ZKP, the smart-card proves to the application that it used the «correct» high-entropy one-time value without revealing it.

But it has two drawbacks:

• allows to protect against the *semi-trusted* smart-card only;

• not secure if the smart-card can terminate the signing process with the error (attack was proposed).
**Solution**

**Idea of the attack:** smart-card completes the signing process successfully only if certain bit of signature equals certain bit of signing key.

\[
\begin{align*}
&u \\
&k \leftarrow \mathbb{Z}_q \\
&r = kP \cdot x \\
&s = ke + dr \\
\text{If } s_0 = d_{r_0} \\
&\quad \text{return } (r, s) \\
&\text{return error}
\end{align*}
\]

- \( s_0 \) – the lowest bit of \( s \)
- \( r_0 \) – the lowest byte of \( r \)
- \( d_i \) – the \( i \)-th bit of \( d \)
1. Motivation
2. Related work
3. Blind signatures
4. Our contribution
5. Open problems
(Conventional) signature scheme

Sig (Signature) scheme:

- \((sk, pk) \leftarrow \text{Sig. KGen( )}: \) key generation algorithm
- \(\sigma \leftarrow \text{Sig. Sign}(sk, m) : \) signing algorithm
- \(b \leftarrow \text{Sig. Vf}(pk, m, \sigma) : \) signature verification algorithm

(Standard) security notion: unforgeability under chosen message attack (UF-CMA)
BS (Blind Signature) scheme:

- \((sk, pk) \leftarrow \text{BS.KGen( )}: \text{key generation algorithm}\)
- \((b', \sigma) \leftarrow \langle \text{BS.Signer}(sk), \text{BS.User}(pk, m)\rangle: \text{interactive signing protocol}\)
- \(b \leftarrow \text{BS.Vf}(pk, m, \sigma): \text{signature verification algorithm}\)
Definition
The BS scheme is a blind version of the Sig scheme, if the KGen and Vf algorithms of these schemes coincide and for any \((sk, pk)\), any message \(m\) and any signature \(\sigma\)

\[
\Pr[(1, \sigma) \leftarrow \langle \text{BS. Signer}(sk), \text{BS. User}(pk, m) \rangle] = \Pr[\sigma \leftarrow \text{Sig. Sign}(sk, m)]
\]

where the corresponding probability spaces are determined by the randomness used in the signing protocol and signing algorithm.

UF-CMA of Sig \(\leftrightarrow\) UF-CMA of BS
**Example**

GOST signature scheme

\[ \text{Sign}(d, m) \]
\[ k \leftarrow \mathbb{Z}_q^* \]
\[ R \leftarrow kP \]
\[ e \leftarrow H(m) \]
\[ r \leftarrow R \cdot x \mod q \]
\[ s \leftarrow ke + dr \]

Camenisch scheme is a blind version of GOST signature scheme

Camenisch blind signature scheme *

\[ \text{Signer}(d) \]
\[ k \leftarrow \mathbb{Z}_q^* \]
\[ R \leftarrow kP \]
\[ e \leftarrow H(m) \]
\[ r \leftarrow R \cdot x \mod q \]
\[ s \leftarrow ke + dr \]

\[ \text{User}(Q, m) \]
\[ \alpha, \beta \leftarrow \mathbb{Z}_q^* \]
\[ R' \leftarrow \alpha R + \beta P \]
\[ r' \leftarrow R' \cdot x \mod q \]
\[ e' \leftarrow H(m) \]
\[ r \leftarrow R \cdot x \mod q \]
\[ e \leftarrow \alpha e' r (r')^{-1} \]

\[ s' \leftarrow s r' r^{-1} + \beta e' \]
\[ \sigma \leftarrow (r', s') \]

Blind signature

Security notions

unforgeability

correct signature can be generated only during the successful interaction with Signer

active adversary - User

blindness

no way to link the \((\text{message, signature})\) pair to the certain execution of the signing protocol

active adversary - Signer
Blind signature

Security notions

unforgeability
- correct signature can be generated only during the successful interaction with Signer
- active adversary - User

blindness
- no way to link the (message, signature) pair to the certain execution of the signing protocol
- active adversary - Signer

Sessions setting:
- Parallel
- Sequential

Signing key:
- Chosen by the adversary
- Honestly generated
Blind signature

Security notions

unforgeability

- correct signature can be generated only during the successful interaction with Signer

active adversary - User

blindness

- no way to link the (message, signature) pair to the certain execution of the signing protocol

active adversary - Signer

Sessions setting:
- Parallel
- Sequential

Signing key:
- Chosen by the adversary
- Honestly generated (wBL)
Outline

1. Motivation
2. Related work
3. Blind signatures
4. Our contribution
5. Open problems
To use blind signature

**Design rationale:** Due to the blindness property, the malicious smart-card learns no information about the signature value, i.e. cannot «control» it.
Formal description of two new security notions for BS:

- **Weak unforgeability** (security against external adversary, $wUNF$): unforgeability against honest-but-curious (passive) User

- **Backdoor resilience** (security against adversary with agent, $BDres$): unforgeability in presence of backdoors in Signer
Our contribution

- **Weak unforgeability (wUNF):**
  unforgeability against honest-but-curious User

---

**Exp_{\text{BS}}^{\text{wUNF}}(A)**

1: *(sk, pk) ← BS.KGen()*
2: \( \mathcal{L} ← \emptyset \)
3: \((m, \sigma) ← A^{\text{Sign}}(pk)\)
4: if \((m, \sigma) ∈ \mathcal{L} : \text{return } 0\)
5: \text{return } BS.Vf(pk, m, \sigma)

---

**Oracle Sign(m)**

1: \((1, (\sigma_{\text{view}})) ← \langle BS.Signer(sk), BS.User(pk, m)\rangle\)
2: \(\mathcal{L} ← \mathcal{L} \cup \{(m, \sigma)\}\)
3: \text{return } \sigma_{\text{view}}
Our contribution

- **Backdoor resilience (BDres):** unforgeability in presence of backdoors in Signer

\[
\text{Exp}_{BS}^{\text{BDres}_k}(A = (A_1, A_2))
\]

1: \( (\text{sk}, \text{pk}) \leftarrow BS.KGen() \)
2: \( \mathcal{L} \leftarrow \emptyset \)
3: \( \text{lost} \leftarrow \text{false} \)
4: \( st \leftarrow A_1(\text{sk}, \text{pk}) \)
5: \( (m, \sigma) \leftarrow S_{A_2}^{\text{Sign}}(\text{pk}) \)
6: \( \text{if } ((m, \sigma) \in \mathcal{L}) \lor (\text{lost} = \text{true}) : \)
7: \( \text{return } 0 \)
8: \( \text{return } BS.Vf(\text{pk}, m, \sigma) \)

\[
\text{Oracle } \text{Sign}(m)
\]

1: \( i \leftarrow 0 \)
2: \( \text{do} \)
3: \( (st, \sigma) \leftarrow (A_1(st), BS.User(\text{pk}, m)) \)
4: \( i \leftarrow i + 1 \)
5: \( \text{until } (i \geq k) \lor (\sigma \neq \bot) \)
6: \( \text{if } \sigma = \bot : \)
7: \( \text{lost} \leftarrow \text{true} \)
8: \( \text{return } \bot \)
9: \( \mathcal{L} \leftarrow \mathcal{L} \cup \{(m, \sigma)\} \)
10: \( \text{return } \sigma \)
Our contribution

**Theorem 1 (informal).** If the BS scheme is wBL- and UF-CMA-secure, then it is BDres-secure.

\[
\text{wBL + UF-CMA} \rightarrow \text{BDres}
\]
The Camenisch blind signature scheme is

- the blind version of GOST: UF-CMA of Sig ↔ UF-CMA of BS;
The Camenisch blind signature scheme is

- the blind version of GOST: UF-CMA of Sig ↔ UF-CMA of BS;
- perfectly wBL-secure (proven by Camenisch);
The Camenisch blind signature scheme is

✓ the blind version of GOST: UF-CMA of Sig ↔ UF-CMA of BS;
✓ perfectly \textit{wBL}-secure (proven by Camenisch);

\begin{align*}
\text{UF-CMA of Sig} & \rightarrow \text{BDres of BS} \quad \text{(by Theorem 1)}
\end{align*}
The Camenisch blind signature scheme is

- the blind version of GOST: UF-CMA of Sig ↔ UF-CMA of BS;
- perfectly $wBL$-secure (proven by Camenisch);
- $wUNF$-secure if GOST is UF-CMA-secure (proven in our work).

$\{\text{UF-CMA of Sig } \leftrightarrow \text{ BDres of BS (by Theorem 1)}\}$
Applying results to GOST

The Camenisch blind signature scheme is

- the blind version of GOST: UF-CMA of Sig ↔ UF-CMA of BS;
- perfectly wBL-secure (proven by Camenisch);
- wUNF-secure if GOST is UF-CMA-secure (proven in our work).

Thus, if GOST is UF-CMA-secure, then the Camenisch scheme is BDres- and wUNF-secure.
Applying results to GOST

The Camenisch blind signature scheme is

- the blind version of GOST: UF-CMA of $\text{Sig} \leftrightarrow$ UF-CMA of BS;
- perfectly wBL-secure (proven by Camenisch);
- wUNF-secure if GOST is UF-CMA-secure (proven in our work).

Thus, if GOST is UF-CMA-secure, then the Camenisch scheme is BDres- and wUNF-secure.

What does it mean for practice?

In order to provide security against backdoors in smart-cards and memory leak in application in case of using GOST, it is enough to implement the Camenisch blind signature scheme.
1. Motivation
2. Related work
3. Blind signatures
4. Our contribution
5. Open problems
Extending security notion

Fully active adversary on the application side (untrusted application or no password-based protection in case of smart-card theft).

Goal: to make a forgery, in particular, by key recovery
The Camenisch blind signature scheme and unforgeability with **active adversary**:

- is not secure in parallel sessions setting
  
  CTCrypt’2022 “On the (im)possibility of secure ElGamal blind signatures”

- **potentially secure** in sequential sessions setting (enough for smart-card case)
  
  positive results for the Schnorr blind signature and its modifications
Thank you for your attention!

lah@cryptopro.ru