Two-Head Dragon Protocol Preventing Cloning of Signature Keys

Przemysław Błaśkiewicz, Przemysław Kubiak, Mirosław Kutyłowski

Wrocław University of Technology

INTRUST 2010, Beijing, 14.12.2010

Security threats for private keys on a smart card

Main concerns:

- keys generated on the card: quality of randomness on a smart card might be insufficient,
- keys generated by the service provider: key copies out of control of a signer,
- key leakage by side channel analysis,
- malicious implementation (e.g. kleptographic leakage of private keys via signatures or public keys).

Smart cards certification:

Certification of the product

- increasingly complex and costly,
- users must trust certification bodies,
- are the certified and the delivered products the same?
  (it is infeasible to inspect tamper-proof devices)

Another approach

- Make evaluation of the product easier for the end-user.
- Move responsibility and internal tests to the manufacturer.

Another approach

- Make evaluation of the product easier for the end-user.
- Move responsibility and internal tests to the manufacturer.

Thus

- Verify behavior also at the protocol level (examples: tamper evidence protocols, e-voting systems).
- At least two mechanisms possible:
  - detection of misbehavior (e.g. a central server periodically changing internal state of smart cards)
  - imposing penalty on the card manufacturer (Two-Head Dragon),

Assumptions:

- We assume that an adversary is able to get **all secret keys** present on the smart-card (unlike for fail-stop protocols).
- If the signature keys are used by the adversary, then they should become publicly known and the owner of the smart card **may effectively deny all signatures** made.
- Hence, there is **no reason to forge a signature by an adversary**.

The Idea of Two-Head Dragon

The Main Idea:

Some magic ..

- We ask a dragon to execute all cryptographic operations on the smart-card.

The Main Idea:

Some magic ..

- We ask a dragon to execute all cryptographic operations on the smart-card.
- Apart from creating signatures, a dragon is guarding fair use of signature keys.

The Main Idea:

Some magic ..

- We ask a dragon to execute all cryptographic operations on the smart-card.
- Apart from creating signatures, a dragon is guarding fair use of signature keys.
- A dragon has two heads.

The Main Idea:

Some magic ..

- We ask a dragon to execute all cryptographic operations on the smart-card.
- Apart from creating signatures, a dragon is guarding fair use of signature keys.
- A dragon has two heads.
- Each time when we ask for a signature, one of the heads responds.

The Main Idea:

Some magic ..

- We ask a dragon to execute all cryptographic operations on the smart-card.
- Apart from creating signatures, a dragon is guarding fair use of signature keys.
- A dragon has two heads.
- Each time when we ask for a signature, one of the heads responds.
- The answer is not only a signature, but also a half of some incantation related to the signature.

The Main Idea:

Some magic ..

- We ask a dragon to execute all cryptographic operations on the smart-card.
- Apart from creating signatures, a dragon is guarding fair use of signature keys.
- A dragon has two heads.
- Each time when we ask for a signature, one of the heads responds.
- The answer is not only a signature, but also a half of some incantation related to the signature.
- A half of an incantation has no magical effect.

The Main Idea:

.. Some magic

- The situation changes if two dragons get the same cryptographic keys.

The Main Idea:

.. Some magic

- The situation changes if two dragons get the same cryptographic keys.
- In fact, as long as only one dragon is asked, nothing happens.

The Main Idea:

.. Some magic

- The situation changes if two dragons get the same cryptographic keys.
- In fact, as long as only one dragon is asked, nothing happens.
- If two dragons are asked the same question, then it might happen that one dragon says the left side of the incantation and the another dragon says the right side of the incantation.

The Main Idea:

Some magic

- The situation changes if two dragons get the same cryptographic keys.
- In fact, as long as only one dragon is asked, nothing happens.
- If two dragons are asked the same question, then it might happen that one dragon says the left side of the incantation and the another dragon says the right side of the incantation.
- **If both parts of the incantation are said the magic starts to work: all signatures created with these keys get burned.**

Example Realization

not in the pre-proceedings

System components

- Probabilistic signature scheme C_{Prob} (for signing messages).
- Rabin-Williams signatures RW (for incantations).
- Incantations are square roots: two square roots from the same value having different Jacobi symbol reveal the private key, i.e. factorization of the modulus.
- A one-way counter (for asking questions to the dragon). The counter might be implemented as a hash-chain.

Setup phase

During deployment, apart from generating the public and private keys for the two signature schemes and generating a hash chain, the ID-card is bounded to make the following dependence:

> If the secret key of RW-signature scheme is revealed, then the secret key of the probabilistic scheme becomes publicly known as well.

Signature generation ..

Creating a signature for a message M ..

1. In order to sign a message M the card receives a next portion of consecutive counter values (say 100 values) $t_1, \ldots, t_{100}$. (We have $t_{i-1} = h(t_i)$, and the card checks correctness of values t_i).

Signature generation ..

Creating a signature for a message M ..

1. In order to sign a message M the card receives a next portion of consecutive counter values (say 100 values) $t_1, \ldots, t_{100}$. (We have $t_{i-1} = h(t_i)$, and the card checks correctness of values t_i).
2. Hash value $H(M)$ of M is calculated, let $b_1, \ldots, b_{100}$ be the last 100 bits of the hash.

Signature generation ..

Creating a signature for a message M ..

1. In order to sign a message M the card receives a next portion of consecutive counter values (say 100 values) $t_1, \ldots, t_{100}$. (We have $t_{i-1} = h(t_i)$, and the card checks correctness of values t_i).
2. Hash value $H(M)$ of M is calculated, let $b_1, \ldots, b_{100}$ be the last 100 bits of the hash.
3. For each value $t_1, \ldots, t_{100}$ its square root s_i, i.e. its RW signature, is calculated by the ID-card. Required value of Jacobi symbol of the square root s_i is indicated by b_i (i.e. for each t_i half of incantation is indicated by the message M).
   (This step is costly).

.. Signature generation

.. creating a signature for a message M

4. Concatenation of $H(M)$, value t_{100}, and sequence $S = s_1, \ldots, s_{100}$ is signed with the probabilistic scheme C_{Prob}. The signature is:

$$C_{Prob}(H(M)||t_{100}||S), t_{100}, S$$

Signature verification

$$C_{Prob}(H(M)||t_{100}||S), t_{100}, S$$

Anyone can check the following conditions:

1. Is t_{100} a value from the hash chain assigned to the user's certificate?
2. Is s_i a RW-signature of t_i, $i = 1 \ldots, 100$?
3. Has s_i the value of Jacobi symbol indicated by bit b_i from the tail part of $H(M)$?
4. Is $C_{Prob}(H(M)||t_{100}||S)$ a valid signature under $H(M)||t_{100}||S$?

Prevention of usage of leaked keys

- In order to create a signature of M' , an adversary must use some 100 consecutive values $t'_1, \ldots, t'_{100}$ from user's hash chain.

Prevention of usage of leaked keys

- In order to create a signature of M' , an adversary must use some 100 consecutive values $t'_1, \ldots, t'_{100}$ from user's hash chain.
- Message M' to be signed determines a sequence of bits $b'_1, \ldots, b'_{100}$.

Prevention of usage of leaked keys

- In order to create a signature of M' , an adversary must use some 100 consecutive values $t'_1, \ldots, t'_{100}$ from user's hash chain.
- Message M' to be signed determines a sequence of bits $b'_1, \ldots, b'_{100}$.
- The bits indicate halves of incantations (appropriate square roots) for the corresponding t'_i.

Prevention of usage of leaked keys

- In order to create a signature of M' , an adversary must use some 100 consecutive values $t'_1, \ldots, t'_{100}$ from user's hash chain.
- Message M' to be signed determines a sequence of bits $b'_1, \ldots, b'_{100}$.
- The bits indicate halves of incantations (appropriate square roots) for the corresponding t'_i.
- To make factorization of the modulus publicly known it suffices that for one i the bit b'_i (i.e. indication of value of the Jacobi symbol of the square root) is different from the bit calculated by the original card for hash value t'_i.

Prevention of usage of leaked keys

- In order to create a signature of M' , an adversary must use some 100 consecutive values $t'_1, \ldots, t'_{100}$ from user's hash chain.
- Message M' to be signed determines a sequence of bits $b'_1, \ldots, b'_{100}$.
- The bits indicate halves of incantations (appropriate square roots) for the corresponding t'_i.
- To make factorization of the modulus publicly known it suffices that for one i the bit b'_i (i.e. indication of value of the Jacobi symbol of the square root) is different from the bit calculated by the original card for hash value t'_i.
- Due to deployment procedure, factoring the modulus used by RW signatures reveals the private key of C_{Prob}.

Chances of the adversary

- To avoid invalidating all signatures (including the forged one) the adversary must modify M' and search for a sequence $t'_1, \ldots, t'_{100}$ such that bits $b'_1, \ldots, b'_{100}$ will agree with those calculated by the original card.

Chances of the adversary

- To avoid invalidating all signatures (including the forged one) the adversary must modify M' and search for a sequence $t'_1, \ldots, t'_{100}$ such that bits $b'_1, \ldots, b'_{100}$ will agree with those calculated by the original card. This is quite unlikely, if the hash chain has length about $100 \cdot 2^{16}$ values and the adversary looks for a collision on 100 bits.

Chances of the adversary

- To avoid invalidating all signatures (including the forged one) the adversary must modify M' and search for a sequence $t'_1, \ldots, t'_{100}$ such that bits $b'_1, \ldots, b'_{100}$ will agree with those calculated by the original card. This is quite unlikely, if the hash chain has length about $100 \cdot 2^{16}$ values and the adversary looks for a collision on 100 bits.
- Calculating a hundred of half-incantations for a single signature of message M is time consuming. But there is an efficient algorithm of this kind as well.

Conclusions

A new paradigm for guarding electronic signatures

- it is hard to guarantee and convince a user that the secret keys are **really** under his sole control,
- ... but now we have methods that **prevent using stolen keys** for signature creation

You may steal my secret keys, but if you use them they become useless.

Thanks for your attention!

This work has been supported by Polish Ministry of Science and Education and Foundation for Polish Science.

FNP Fundacja na rzecz Nauki Polskiej