Dipartimento di Matematica e Informatica
Università di Catania

Off-line/On-line signatures

Theoretical aspects and Experimental Results

Dario Fiore

fiore@dmi.unict.it http://www.dmi.unict.it/~fiore

Joint work with D. Catalano, M. Di Raimondo and R. Gennaro

Outline

- Off-line/On-line signatures
  - Even et al. construction
  - Shamir-Tauman construction
- A unifying paradigm
- Experimental results
  - Implementation notes
  - Analysis of the results

Off-line/On-line signatures

- The signing process is divided in two phases:
- a computationally intensive part is performed off-line *(i.e. before the message being signed is known);*
  - some temporary data ω is produced.
  - ω is used in the on-line phase to compute the actual signature (i.e. when the msg to be signed is known).
- The computation in the on-line phase should require very little effort.

Possible applications

- Such type of signatures can be useful in many contexts:
- mobile devices with reduced computational resources,
- public services with a huge amount of requests
- Time-constrained applications

EGM construction

- The first construction of Off-line/On-line signatures was proposed by [EGM96]
- Basic idea: combine two different types of digital signatures:
  - many-times (or “regular”) signatures
  - one-time signatures.

Regular/One-time signatures

- Regular signatures can be used to sign many messages
- In one-time signatures a private key can be used to sign **only a single message**.
- One-time signatures can be constructed more efficiently

EGM construction

- Let *(KG,Sign,Ver)* a regular signature scheme and *(KG^{ots},$Sign^{ots}$,Ver^{ots})* and OTS scheme
- The signer generates a pair of keys *(VK, SK)*$\leftarrow$*KG()* for a regular signature scheme
- **Off-line**
  - *(vk,sk)*$\leftarrow$*KG^{ots}()*
  - Sign *vk* using *SK*, *S=$Sign_{SK}$(vk)*.
- **On-line**
  - Once *m* is given, compute, *s=$Sign_{sk}$(m)*
  - The final signature for *m* is σ *=(vk, S, s)*.
- **Verification(m, σ)**
  - *Ver^{ots}(vk,m,s) and Ver(VK,vk,S)*

One-way functions-based OTS

- EGM uses general constructions of **one-time signatures (**i.e. from **one-way functions)**
- very fast to compute and verify
- the produced signature is quite long
  - grows quadratically with |m|

Shamir-Tauman construction

- In [ST01] ST cope with the problem of reducing the signature length
- **Basic idea:** combine regular signatures with *chameleon hashing*

Chameleon Hashing

- A chameleon hash function is defined by a public key *pk* and a secret trapdoor *tk*
- The function $C_{pk}(m,r)$ takes in input:
  - a message *m*
  - a random string *r*
- **Property:** C_{pk} is collision resistant unless one knows the trapdoor *tk*.

Finding collisions

- **Example** Given $c=C_{pk}(m,r)$ and an arbitrary different message m'
- the holder of the trapdoor can find r' such that $c = C(m',r')$

- For many chameleon hash functions, this collision-finding procedure is very efficient (i.e. requiring only a single modular multiplication)

ST construction

- Generate *(VK, SK)* keys for a regular signature scheme
- **Off-line**
  - (*pk*,*tk)* key pair for a chameleon hash function
  - Compute *c=C(a,r') // a* msg, *r'* random
  - Compute $S=Sign_{SK}(c)$

ST construction (2)

- **On-line**
  - On input a message *m*, use *tk* to find *r* such that: $c=Cpk(m,r)$
  - Output $\sigma=(pk,r,c,S)$
- **Verification(PK, m, σ)**
  - Verify that $c=C_{pk}(m,r)$ and that $S=Sign_{SK}(c)$
- Such paradigm is also called "***hash-sign-switch***"

A unifying paradigm

- At first glance, the ST approach and the EGM methodology look very different.
- **Question:** Is this actually the case?
  - Can the two approaches be seen as different istantiations of the same paradigm?
  - Yes, if chameleon hash funcs can be seen as OT sigs.

Observations

1. EGM construction remains secure even if one replaces standard one-time signatures with simpler ones that we call *oblivious one time signatures*
2. We prove that chameleon hash functions are a form of oblivious one-time signatures

OTS security definition

- *(KG,Sign,Ver)* is a secure one-time signature scheme if for every efficient forger *F*, the following is negligible in *l*:

$$\Pr\left[\begin{array}{l}(vk,sk) \leftarrow KG(1^l);\\ M \leftarrow F(vk);\\ \sigma \leftarrow Sign(sk,M);\\ (M',\sigma') \leftarrow F(vk,M,\sigma):\\ Ver(vk,M',\sigma') = 1 \wedge M' \neq M\end{array}\right]$$

Oblivious OTS

$$\Longrightarrow \Pr\left[\begin{array}{l} M \leftarrow F(); \\ (vk,sk) \leftarrow KG(1^l); \\ \sigma \leftarrow Sign(sk,M); \\ (M',\sigma') \leftarrow F(vk,M,\sigma): \\ Ver(vk,M',\sigma') = 1 \wedge M' \neq M \end{array}\right]$$

Chameleon Hash-based oOTS

- KeyGeneration
  - Produce *(pk,tk)*, a Cham.Hash key pair
  - Compute $c=C_{pk}(\alpha, r)$ // *msg, r random*
  - oOTS public key is *(pk,c)*
  - oOTS signing key is *(tk, α, r)*
- $\text{Sign}_{tk}(m)$
  - Use *tk* to find *s* such that $c=C_{pk}(m,s)$
  - The signature is *(m,s)*
- $\text{Verify}_{pk}(m,s)$
  - Check if $c=C_{pk}(m,s)$

Theorem (informal)

- *The proposed construction is an obliviously secure OTS if the underlying primitive is a chameleon hash function*

Experimental results

- We implemented several instatiations of the EGM and ST paradigms
- Each instantiation used different combinations of regular signature schemes and OTS/Chameleon hash schemes

Implemented schemes

- Regular signature schemes
  - Gennaro-Halevi-Rabin (GHR)
  - Cramer-Shoup (CS)
- OTS
  - One-way functions-based OTS were implemented using a truncated output of SHA-1 (the first l bits according to a security parameter l)

Implemented schemes

- Chameleon Hash functions
  - Discrete Log-based
  - RSA-based
- Message Hashing
  - Fully Collision Resistant (FCR)
    - *SHA-1(m)*
  - Target Collision Resistan (TCR)
    - $Trunc_l(SHA\text{-}1(m \oplus k))$ for a random key *k* signed together with *m*

Hash functions

- When signing messages one usually hashes them down with a Fully Collision Resistant (FCR) function to shorten them (*i.e. SHA-1*)
- One can use a Target Collision Resistant (TCR) hash function provided that the key of the hash function is signed together with the message digest and sent as part of the signature
- **Advantage:** TCR message digest may be shorter
- **Drawback:** we need to sign the key

Analysis of the results

- EGM *vs* ST
  - The use of TCR hashing leads to somewhat comparable results
  - Minimum ST on-line signing time: 0.03ms
  - Minimum EGM on-line signing time: 0.47ms
  - Signature length (in bits): 2144(EGM) vs 1184(ST)

Analysis of the results

- GHR *vs* CS
  - GHR outperforms CS in almost all parameters: off-line and on-line signing time, and signature size.
  - CS is faster only in verification time,
    - Not surprising, as GHR must use longer verification exponents.

Analysis of the results: Chameleon Hashing

- DL-based *vs* RSA-based
  - The time required for hash evaluation is comparable in both the schemes
  - But the DL-based one has a notable advantage in the collision finding step
    - 0.03ms *vs* 11ms!
  - This operation is fundamental because it is in the on-line phase

Analysis of the results: TCR hashing

- It has a dramatic impact on the efficiency of OTS schemes (and thus EGM)
- Still it improves ST because it reduces size of exponents (mainly in RSA)

Main References

- [1] S. Even, O. Goldreich and S. Micali, ***On-Line/Off-Line Digital Signatures***, *Journal of Cryptology, vol. 9, n. 1, pp. 35-67, Springer, 1996*
- [2] A. Shamir and Y. Tauman, **Improved On-line/Off-line Signature Schemes** *Advances in Cryptology – proceedings of CRYPTO'01, LNCS 2139, Springer-Verlag, pp.355-367, 2001.*
- [3] D.Catalano, M.Di Raimondo, D.Fiore and R. Gennaro. ***Off-Line/On-Line Signatures: Theoretical aspects and Experimental Results*** *Proceedings of PKC 2008 - to appear*

✦**Thanks!**

✦Questions…