Optimally Efficient Security Proofs for Hash-then-Publish Time-stamping

Margus Niitsoo
(with Ahto Buldas)

University of Tartu / Cybernetica AS

July 6, 2010
Outline

1. Timestamping
2. Proving it Secure
What is timestamping

- What patent offices do
- Malicious patent clerk problem
A time stamping scheme

- Idea: publish a small hash value in the paper every day
- Formally 3 parties: Client, Server, Repository
- Clients generate fixed sized signatures from their documents
- Server takes all these signatures and commits something to the repository.
- Server then gives certificates to all the Clients which prove that their document signatures are indeed tied to the value committed.
A time-stamping scheme

On the server side
- Essentially one would like a multi-value binding commitment scheme
- We have to be able to decommit one element at a time
- Also, we are more worried about some things than others.
- There are trivial solutions that take a lot of memory
Harber and Stornetta proposed the following (CRYPTO 90):

- Clients generate fixed sized signatures from their documents.
- Server constructs a tree from signatures and uses a hash function $h_s : \{0, 1\}^{2n} \rightarrow \{0, 1\}^n$ to calculate the root.
The Scheme of Harber and Stornetta
The Scheme of Harber and Stornetta

- Clients generate fixed sized signatures from their documents.
- Server constructs a tree from signatures and uses a hash function $h_s : \{0, 1\}^{2n} \rightarrow \{0, 1\}^n$ to calculate the root.
- Certificate for $x$ is just a path in the tree from $x$ to root:
  - Contains the structure of the path – left or right turn.
  - Contains the other inputs needed to get to the root.
The Scheme of Harber and Stornetta

- Clients generate fixed sized signatures from their documents
- Server constructs a tree from signatures and uses a hash function \( h_s : \{0, 1\}^{2n} \rightarrow \{0, 1\}^n \) to calculate the root
- Certificate for \( x \) is just a path in the tree from \( x \) to root
  - Contains the structure of the path – left or right turn
  - Contains the other inputs needed to get to the root
- The shape of the tree needs to be restricted for meaningful results.
Security criterion – backdating a previously unknown document (Buldas, Saarepera, ASIACRYPT 2004)

**Definition**

A TS scheme is secure if for every unpredictable \((A_1, A_2)\):

\[
\Pr \left[ (r, a) \leftarrow A_1(1^k), (x, c) \leftarrow A_2(r, a) : \text{Ver}(x, c, r) = 1 \right] = k^{-\omega(1)}. \tag{1}
\]
Cryptographic practice

- Nearly nothing is unconditionally secure.
- Everything built on assumptions
- The simpler and the more widespread, the better
Collision resistant (hash) function

Assume we have an $h$ for which it is hard to find two inputs $x \neq x'$ such that $h(x) = h(x')$.

Such a pair is called a collision.

Such functions are assumed to exist.
Collision Extraction Property

Definition

Let $\text{Ver}^h$ be a verification procedure and $c \mapsto \rho(c) \in \mathcal{N}$ be a shape function such that if $\text{Ver}^h(x_1, c_1, r) = \text{Ver}^h(x_2, c_2, r)$, $\rho(c_1) = \rho(c_2)$, and $(x_1, c_1) \neq (x_2, c_2)$, then the $h$-calls of $\text{Ver}^h(x_1, c_1, r)$ and $\text{Ver}^h(x_2, c_2, r)$ comprise an $h$-collision.
The Scheme of Harber and Stornetta

Margus Niitsoo (with Ahto Buldas)

Optimally Efficient Security Proofs for Hash-then-Publish Time-stamping
The Easy Proof

- Assume $A = (A_1, A_2)$ breaks time-stamping with prob $\delta$.
- Call $A_1$ once (to get commitment)
- Call $A_2$ twice to get $x_1, x_2$ along with certificates
• Assume $A = (A_1, A_2)$ breaks time-stamping with prob $\delta$.
• Call $A_1$ once (to get commitment)
• Call $A_2$ twice to get $x_1, x_2$ along with certificates
• Succeeds with probability $\frac{\delta^2}{N}$. 

Margus Niitsoo (with Ahto Buldas)
The Easy Proof is not Good Enough

- Succeeds with probability $\frac{\delta^2}{N}$.
  - $2^{26}$ stamps per second with a lifetime of 34 years.
- Assume $h$ secure to the birthday bound
- Require meaningful security
Not Good Enough in what respect

- Require meaningful security
- Any $\delta$ can be amplified by repeated calls to $A$.
  - At a cost in running time
- Makes sense to consider Time-success ratio
  - Expected running time divided by success probability
  - Average time it takes to break the primitive
- In the previous case $\frac{t'}{\delta^7} \approx 2N \frac{t}{\delta^2}$.
  - Quadratic in time-success ratio
The Easy Proof is not Good Enough

- Succeeds with probability $\frac{\delta^2}{N}$.
- We want a global-scale time-stamping service ($N = 2^{80}$)
  - $2^{26}$ stamps per second with a lifetime of 34 years.
- Assume $h$ secure to the birthday bound
- Require meaningful security
Succeeds with probability $\frac{\delta^2}{N}$.

We want a global-scale time-stamping service ($N = 2^{80}$)
  - $2^{26}$ stamps per second with a lifetime of 34 years.

Assume $h$ secure to the birthday bound

Require meaningful security
  - $2^{80}$ seconds to break on average
The Easy Proof is not Good Enough

- Succeeds with probability $\frac{\delta^2}{N}$.
- We want a global-scale time-stamping service ($N = 2^{80}$)
  - $2^{26}$ stamps per second with a lifetime of 34 years.
- Assume $h$ secure to the birthday bound
- Require meaningful security
  - $2^{80}$ seconds to break on average
- All this implies output length of 370 bits in the worst case.
The Easy Proof is not Good Enough

- Succeeds with probability $\frac{\delta^2}{N}$.
- We want a global-scale time-stamping service ($N = 2^{80}$)
  - $2^{26}$ stamps per second with a lifetime of 34 years.
- Assume $h$ secure to the birthday bound
- Require meaningful security
  - $2^{80}$ seconds to break on average
- All this implies output length of 370 bits in the worst case.
- We REALLY want to use a 256 bit function.
Provide a reduction where $\frac{t'}{\delta'} \approx 48\sqrt{N} \frac{t}{\delta^2}$.
Provide a reduction where $\frac{t'}{\delta'} \approx 48\sqrt{N} \frac{t}{\delta^2}$.

- Considerably better for large $N$.
- Still quadratic in Time-success ratio.
Show a lower bound on the reduction efficiency (using oracle separation)

- Power-1.5 security loss in time-success ratio.
- Question: can it be achieved?
In short: YES

- Our reduction has \( \frac{t'}{\delta'} \approx 14 \sqrt{N} \frac{t}{\delta^{1.5}} \).
In short: YES

- Our reduction has \( \frac{t'}{\delta'} \approx 14\sqrt{N} \frac{t}{\delta^{1.5}} \).
- Adversary construction basically same, just more \( A_2 \) queries and more precise analysis.
- Allows us to prove that 256 bit hash functions are enough for the described setting.
- Also – first optimality result in time-success ratio.
Thank You!

Any questions are welcome!