Outsourced Storage & Proofs of Retrievability

Hovav Shacham, UC San Diego
Brent Waters, SRI International

The Setting

- Client stores (long) file with server
  - Wants to be sure it's actually there
- Motivation: online backup; SaaS
- Long-term reliable storage is expensive

Example Protocols

$(h = h(M))$ $\mathcal{V}$ $\mathcal{P}$ (M)

$$\xleftarrow{\quad\quad\quad\quad\quad\quad M}$$

$h \stackrel{?}{=} h(\cdot)$

Kotla, Alvisi, Dahlin, Usenix 2007:

(M) $\mathcal{V}$ $\mathcal{P}$ (M)

$$\xrightarrow{c\quad\quad\quad\quad\quad\quad}$$

$$\xleftarrow{\quad\quad\quad\quad h(c\|M)}$$

$h(c\|M) \stackrel{?}{=} \cdot$

How do we evaluate protocols of this sort?

Systems Criteria

- Efficiency:
  - Storage overhead
  - Computation (including # block reads)
  - Communication
- Unlimited use
- Stateless verifiers
- Who can verify? File owner? anyone?

Crypto criterion

- Only an adversary storing the file can pass the verification test
- Possible to extract M from any prover P' via black-box access
- (Cf. ZK proof-of-knowledge)

- Insight due to Naor, Rothblum, FOCS 2005 and Juels, Kaliski, CCS 2007

Security Model – I

- Keygen: output secret key *sk*
- Store (*sk*, file *M*):
  output tag *t*, encoded file *M**
- Proof-of-storage protocol:

$$\{0,1\} \stackrel{R}{\leftarrow} \left(\mathcal{V}(sk, t) \rightleftharpoons \mathcal{P}(t, M^*)\right)$$

- Public verifiability:
  - Keygen outputs keypair (*pk*,*sk*)
  - Verifier algorithm takes only *pk*

Security Model – II

- Challenger generates *sk*
- Adversary makes queries:
  - "store M_i" ⇒ get t_i, M_i^*
  - "protocol on t_i" ⇒ interact w/ $V(sk, t_i)$.
- Finally, adversary outputs:
  - challenge tag t from among $\{t_i\}$
  - description of cheating prover P' for t

Security Model — III

- Security guarantee:

  $\exists$ extractor algorithm Extr st. when

  $$\Pr\left[\left(\mathcal{V}(\mathrm{sk}, \mathrm{t}) \rightleftharpoons \mathcal{P}'\right) = 1\right] \geq \epsilon$$

  we have

  $$\mathrm{Extr}(\mathrm{sk}, \mathrm{t}, \mathcal{P}') = M$$

  except with negligible probability

Probabilistic Sampling

- Want to check 80 blocks at random, not entire file
- Pr[detect 1-in-10^6 erasure]: < 0.01%
- Pr[detect 50% erasure]: 1 - $(1/2)^{80}$
- So: encode $M \Rightarrow M^*$ st. any 1/2 of blocks suffice to recover M: erasure code
- Due to Naor, Rothblum, FOCS 2005

The Simple Solution

- Store:
  - erasure encode $M \Rightarrow M^*$
  - for each block m_i of M^*,
    store authenticator $\sigma_i = \mathrm{MAC}_k(i, m_i)$
- Proof of storage:

$(k) \quad \mathcal{V} \qquad\qquad \mathcal{P} \quad \left(\{(m_i, \sigma_i)\}_{i=1}^{n}\right)$

$I \overset{R}{\subseteq} [1, n] \quad (|I| = 80)$ $\longrightarrow$

$\longleftarrow$ $\{(m_i, \sigma_i)\}_{i \in I}$

$$\sigma_i \overset{?}{=} MAC_k(i, m_i)$$

Lower communication using homomorphic authenticators

Improved Solution (Try #1)

- Downside to simple solution:
  response is 80 blocks, 80 authenticators
- Let's send Σm_i instead!

(k) $\quad \mathcal{V}$ $\qquad\qquad\qquad\qquad$ $\mathcal{P}$ $\quad \left(\{(m_i, \sigma_i)\}_{i=1}^{n}\right)$

$$I \overset{R}{\subseteq} [1, n] \quad (|I| = 80) \longrightarrow$$

$$\longleftarrow \mu = \sum_{i \in I} m_i \quad \sigma = \sum_{i \in I} \sigma_i$$

Improved Solution (Try #1)

- Downside to simple solution:
  response is 80 blocks, 80 authenticators
- Let's send Σm_i instead!

$$(k) \qquad \mathcal{V} \qquad\qquad\qquad \mathcal{P} \quad \left(\{(m_i, \sigma_i)\}_{i=1}^{n}\right)$$

$$I \overset{R}{\subseteq} [1, n] \quad (|I| = 80)$$

$$\longrightarrow$$

$$\mu = \sum\nolimits_{i \in I} m_i \quad \sigma = \sum\nolimits_{i \in I} \sigma_i$$

???

$$\longleftarrow$$

Homomorphic Authenticators

- Problem: have linear combination of messages m_i
- Need to authenticate via some function of $\{\sigma_i\}$
- Ateniese et al., CCS 2007:
  RSA-based homomorphic authenticators;
  $\prod_i \sigma_i^{\nu_i}$ authenticates $\sum_i \nu_i m_i$

Our Contributions

1. Efficient homomorphic authenticators based on PRFs and on bilinear groups
2. A full proof for (improved) simple protocol, against *arbitrary* adversaries

PRF Authenticator

- PRF f: $\{0,1\}^* \rightarrow K$; $m_i \in K$; K: GF(2^{80}) or Z_p
- Keygen: PRF key k; $\alpha \in K$
- Authenticate: $\sigma_i \leftarrow f_k(i) + \alpha \cdot m_i$
- Aggregate:

$$\sigma \leftarrow \sum \nu_i \sigma_i \quad \text{and} \quad \mu \leftarrow \sum \nu_i m_i$$

- Verify:

$$\sigma \stackrel{?}{=} \sum \nu_i f_k(i) + \alpha\mu$$

BLS Authenticator

- Bilinear map e: $G_1 \times G_2 \rightarrow G_T$, $\langle u \rangle = G_1$.
- Keygen: sk: $x \in \mathbf{Z}_p$; pk: $v = g_2^x \in G_2$.
- Authenticate: $\sigma_i \leftarrow \left[H(i)u^{m_i}\right]^x$
- Aggregate:

$$\sigma \leftarrow \prod \sigma_i^{\nu_i} \quad \text{and} \quad \mu \leftarrow \sum \nu_i m_i$$

- Verify:

$$e(\sigma, g) \stackrel{?}{=} e\Big(u^{\mu} \cdot \prod H(i)^{\nu_i}, v\Big)$$

Improved Solution (Try #2)

(k, α) $\quad \mathcal{V}$ $\qquad\qquad\qquad\qquad$ $\mathcal{P}$ $\quad \left(\{(m_i, \sigma_i)\}_{i=1}^{n}\right)$

$I \stackrel{R}{\subseteq} [1, n] \quad (|I| = 80)$

$\nu_i \stackrel{R}{\leftarrow} K \quad \text{for } i \in I$

$$\xrightarrow{Q = \{(i, \nu_i)\}}$$

$$\mu \leftarrow \sum_{(i,\nu_i) \in Q} \nu_i m_i$$

$$\sigma \leftarrow \sum_{(i,\nu_i) \in Q} \nu_i \sigma_i$$

$$\xleftarrow{\mu, \sigma}$$

$$\sigma \stackrel{?}{=} \sum_{(i,\nu_i) \in Q} \nu_i f_k(i) + \alpha\mu$$

Communication & storage

- PRF solution: 80-bit μ, 80-bit σ
- BLS solution: 160-bit μ, 160-bit σ
- But: 100% storage overhead
- Storage/communication tradeoff:
  - split each block into s sectors
  - one authenticator per block:
    - response: $(1+s)\times 80$ bits [or $\times 160$ bits]
    - storage overhead: $1/s$

The proof of security

Security Proof Outline

1. "Straitening": whenever (μ,σ) verify correctly, μ was computed as $\Sigma v_i m_i$
2. "Extraction": can extract 1/2 of blocks from prover P' that outputs $\mu=\Sigma v_i m_i$ on ε-fraction of queries, $\perp$ otherwise
3. "Decoding": recover M from any 1/2 of M^* blocks

Attack on Improved Solution Try #1

- Attacker picks index i^*
- For $i \neq i^*$, sets $a_i \leftarrow \pm 1$, stores $m' \leftarrow m_i + a_i m_{i^*}$
- for query I st. $i^* \notin I$, compute

$$\mu' = \sum_{i \in I} m'_i = \sum_{i \in I} (m_i + a_i m_{i^*}) = \mu + m_{i^*} \sum_{i \in I} a_i$$

- this is correct if #(+1) = #(-1) in Σa_i:

$$\Pr\left[0 = \sum\nolimits_{i \in I} a_i\right] = \binom{80}{40} \cdot \frac{1}{2^{80}} \approx 8.89\%$$

Attack (cont.)

Attacker knows dim (*n*-1) subspace:

$$\begin{pmatrix} 1 & & & \cdots & 0 & \pm 1 \\ & 1 & & \ddots & \vdots & \pm 1 \\ & & \ddots & & & \pm 1 \\ \vdots & \ddots & & 1 & & \pm 1 \\ 0 & \cdots & & & 1 & \pm 1 \end{pmatrix}$$

But he doesn’t know any single block!

Conclusion

- Homomorphic authenticators from PRFs, BLS
- "Improved Solution, Try #2":
  - compact response (& query in r.o. model)
  - secure against arbitrary adversarial behavior
- Security requires proof — some okay-looking schemes are insecure

http://cs.ucsd.edu/~hovav/papers/sw08.html