Overview

• BiBa stands for “Bins and Balls”
  – Use one-way functions without trapdoors (e.g., hash functions)
• BiBa signature scheme
• BiBa broadcast authentication protocol

BiBa Signature Scheme

• Precompute SEALs
  – SEAL: SElf Authenticating vaLues
• Signature generation
  – Exploit SEALs and the difficulty of finding collisions under hash functions
• Signature verification
  – Verify SEAL
  – Verify collisions
**SEAL**

- Each SEAL is randomly generated
- Given a SEAL $s$, the signer computes $f_s = F_s(0)$, where $F_s$ is a PRF
  - $f_s$ is the commitment to $s$
  - $f_s$ is authenticated to all possible verifiers (e.g., through a RSA signature or pre-distribution)
- In BiBa, the signer has $t$ pre-computed SEALs
  - SEALs: $s_1, s_2, ..., s_t$
  - All SEALs are authenticated to all verifiers

**BiBa Signature: Intuition**

- Sign message $m$
  - Compute hash $h = H(m)$, where $H$ is a hash function
  - Consider a hash function family $G_h$, whose range is $[0, n-1]$
    - Example: $G_h(i) = G_i(i)$, where $G$ is SHA1
  - Compute $G_h$ for all SEALs $s_1, ..., s_t$
    - That is, $G_h(s_1), G_h(s_2), ..., G_h(s_t)$
  - Look for a 2-way collision of SEALs
    - $G_h(s_i) = G_h(s_j)$ with $s_i \neq s_j$
    - The pair $<s_i, s_j>$ forms the signature
- Signature verification
  - Compute hash $h = H(m)$
  - Verify $s_i \neq s_j$ and $G_h(s_i) = G_h(s_j)$

**Basic BiBa Scheme**

![Diagram of BiBa Scheme]

- SEALs (SEALs): $s_1, s_2, s_3, ..., s_t$
- Bins (Range of $G_h$): $G_h$
- Signature
Security of BiBa Signature

- Security comes from
  - The difficulty of finding $k$-way collisions for one-way functions
  - The asymmetric property that the signer has more SEALs than the adversary
    - Signer can easily generate the BiBa signatures with high probability while adversary can’t.
- Exploits the birthday paradox
  - Probability that there is at least one collision of the hashes of $t$ random messages is approximately
    $$1 - e^{-t(t-1)/2N}$$
    where $N$ is range of hash function.

Security of BiBa Signature (Cont’d)

BiBa Signature Scheme

- Basic scheme
  - Signer is not guaranteed to find a signature
- BiBa Signature
  - Sign message $m$
    - $h_i = H(m|c)$, where $c$ is a counter starting from 0
    - $c$ is incremented if no signature is found
    - Compute $G_i$ for all SEALs $s_1, ..., s_t$
    - Look for a $k$-way collision of SEALs
  - Verify signature
    - Verify the $k$ SEALs are distinct
    - Verify that they have the same image
BiBa Broadcast Authentication Protocol

- Sender needs to authenticate potentially infinite stream of messages
- Sender can only disclose a small number of SEALs before attacker would have enough time to forge signature
  - Limit the number of messages that can be signed
- Solution
  - SEAL chains
    - Combination of SEALs and TESLA

SEAL Chains

\[
S_{i+1} = F_{S_{i+1}}(K_{i+1})
\]

Limitation

- High receiver computation overhead
  - Most of the SEALs are not used
  - To authenticate a SEAL, each receiver needs to re-compute many SEALs in a one-way SEAL chain
Extension A

- SEAL boundary
  
  SEALs above the boundary are disclosed
  
  SEAL boundary (0, 2, 3, 0, 1, 2)
  
- If attacker slows down the traffic to the receivers, ...
- Packet losses

Extension B

- To tolerate packet losses
  - Add SEAL boundary information to packets
  - More communication overhead, but also more robust
- Receivers still need to know the sending rate
  - Why?
Motivation

- BiBa
  - Fast signature verification, but
  - Signing cost is high
- Authors’ goal
  - Develop a one-time signature scheme with
    - Fast signing and verification

Preliminary: Bijective Function

- Bijective function
  - Each element of input is mapped onto one and only one element in output
  - Each element of output is mapped onto one and only one element in input
  - Intuitively, there is a one-to-one correspondence between elements of the two sets

Bijective Function $S$

- Let $T = \{1, 2, \ldots, t\}$
- $S$ is a bijective function that outputs the $m$-th $k$-element subset of $T$
Initial Scheme: Based on One-way Functions

• Generalization of Bos and Chaum one-time signatures

• Key generation
  – Generate \( t \) numbers of random \( l \)-bit values
  – Let these be the private key: \( \text{SK} = (s_1, \ldots, s_t) \)
  – Compute the public key \( \text{PK} = (v_1, \ldots, v_t) \),
    • where \( v_i = f(s_i) \) and \( f() \) is a one-way function

Signature Generation and Verification

• Sign a \( b \)-bit message \( m \)
  – Use \( S \) to find the \( m \)-th \( k \)-element subset of \( T \): \( \{i_1, \ldots, i_k\} \)
  – The corresponding values \( (s_{i_1}, \ldots, s_{i_k}) \) are the signature of \( M \)

• Verify message \( m \) and its signature \( (s_1', \ldots, s_k') \)
  – Use \( S \) to find the \( m \)-th \( k \)-element subset of \( T \): \( \{i_1', \ldots, i_k'\} \)
  – Verify \( f(s_1') = v_{i_1'}, \ldots, f(s_k') = v_{i_k} \)

Efficiency Analysis

• Key generation
  – Requires \( t \) evaluations of the one-way function
  – Secret key size = \( l^t \)-bits
  – Public key size = \( f^t \)-bits
    • \( f^t \)-length of the one-way function output

• Signature generation
  – Time to find the \( m \)-th \( k \)-element subset of \( T \)
• Verification
  – Time to sign + \( k \) one-way function operations
Security

- Bijective function S
  - Each input corresponds to one and only one output
- Thus, each $b$-bit message $m$ corresponds to a different $k$-element subset of $T$
  - Knowing the signature of one message, an attacker has to invert at least one of the remaining $t-k$ values in the public key to forge another signature

An Option for S

- Algorithm #1: $C(t, k) = C(t-1, k-1) + C(t-1, k)$
  - If the last element of $T$ belongs to the subset, choose $k-1$ elements from the remaining $t-1$ elements
  - Otherwise, choose $k$ elements from the remaining $t-1$ elements
- Input: $(m, t, k)$
- Steps:
  - If $m < C(t-1, k-1)$
    - Add $t$ to output and recur on $(m, k-1, t-1)$
  - Else
    - Add nothing to output and recur on $(m - C(t-1, k-1), k, t-1)$

HORS: Based on Subset-Resilient Functions

- Replace the Bijective function $S$ with a subset-resilient function $H$
  - $S(m)$ has $k$ elements
  - $S$ guarantees that no two distinct messages have the same $k$-element subset of $T$
  - $H(m)$ has at most $k$ elements
  - $H$ guarantees that it is infeasible to find two distinct messages $m_1$ and $m_2$ such that
    - $H(m_2)$ is a subset of $H(m_1)$
### HORS Operations

**Key Generation**
- **Input:** Parameters $L, k$
- **Output:** $PK = (h, r_1, r_2, \ldots, r_k)$

**Signing**
- **Input:** Message $m$ and secret key $SK = (h, r_1, r_2, \ldots, r_k)$
- **Output:** $\sigma = (s, m', m''_1, m''_2)$

**Verifying**
- **Input:** Signing $\sigma = (s, m', m''_1, m''_2)$ and public key $PK = (h, r_1, r_2, \ldots, r_k)$
- **Output:** $\text{success}$ if $h = \text{hash}(m' || m''_1 || m''_2)$

### Comparison with BiBa

<table>
<thead>
<tr>
<th></th>
<th>BiBa</th>
<th>HORS</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>H</strong></td>
<td>is a random oracle</td>
<td>is a subset resilient function</td>
</tr>
<tr>
<td><strong>Signing</strong></td>
<td>2(t) calls to H (signer needs two trials on average)</td>
<td>One call to H</td>
</tr>
<tr>
<td><strong>Verifying</strong></td>
<td>4 calls to H</td>
<td>One call to H</td>
</tr>
</tbody>
</table>