NC STATE UNIVERSITY Computer Science

“The BiBa One-Time Signature and Broadcast Authentication Protocol”

Rich Larsen

April 9, 2003

Overview

- BiBa stands for “Bins and Balls”
- Originally presented in paper by Adrian Perrig at ACM CCS conference in 2001
- BiBa includes both a digital signature scheme and authentication protocol
- BiBa uses one-way functions without trapdoors (eg., hash functions) .

NC STATE UNIVERSITY Computer Science

Design Requirements for Broadcast Authentication Protocols

- Efficient generation and verification of signatures
- Real-time authentication
- Individual message authentication- no buffering of messages
- Robustness to packet loss
- Scalability- protocol should be independent of number of receivers

NC STATE UNIVERSITY Computer Science

Related Work

- Tesla Protocol also proposed by Perrig
- Splits up time into intervals of uniform duration
- Assigns a unique authentication value to be active during each interval
- Delays the release of the key for the current authentication value until after the interval is over
- Disadvantages of Tesla:
  - Requires "strong" time synchronization between sender and all receivers
  - Receivers must buffer some packets (not real-time authentication)

NC STATE UNIVERSITY Computer Science

BiBa Protocol

- According to author, BiBa meets all the desired requirements for broadcast authentication protocols (only known protocol to do so)
- Advantages:
  - Smaller signature size and faster verification than traditional digital signature protocols based on public key algorithms
- Disadvantage:
  - Requires “weak” time synchronization between sender and receivers (i.e., less than Tesla)
  - Moderate overhead for sender to generate the authentication information (can be parallelized)

BiBa Signature Protocol

- Signer precomputes some random values called *SEAL’s* (SElf Authenticating vaLues)
- SEALS are randomly-generated but can be authenticated using a public key
- Given a SEAL *s,* public key is $f_s = F_s(0)$ *where* $F_s(0)$ is a one-way function or “commitment” to s.
- Signer has precomputed *t* seals $s_1, \ldots, s_t$ *and commitments for each SEAL.*
- Receiver knows commitments $F_s(0)$ *for 1*

BiBa Signature Generation Algorithm

- Given message *M*, compute hash $h = H(M||c)$ *where c is a counter starting from 0.*
- G_h is a particular instance from a family of one-way function whose range is 0, n-1 (i.e., *n* possible output values)
- Compute G_h for all seals $s_1, \ldots, s_t$. Each should map to a value between 0 and n-1
- Look for a k-way collision of seals: (e.g., for k=2, look for $G_h(s_i) = G_h(s_j)$ *with* $s_i \,!= s_j$
- The pair $<s_i, s_j>$ forms the signature
- If no k-way collisions occur, increment c and start over

BiBa Signature Generation Scheme

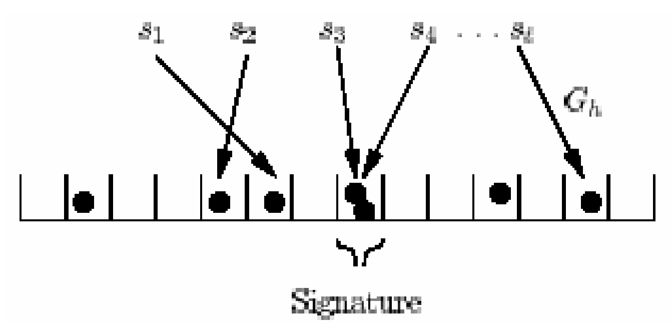


Figure 1: Basic BiBa scheme

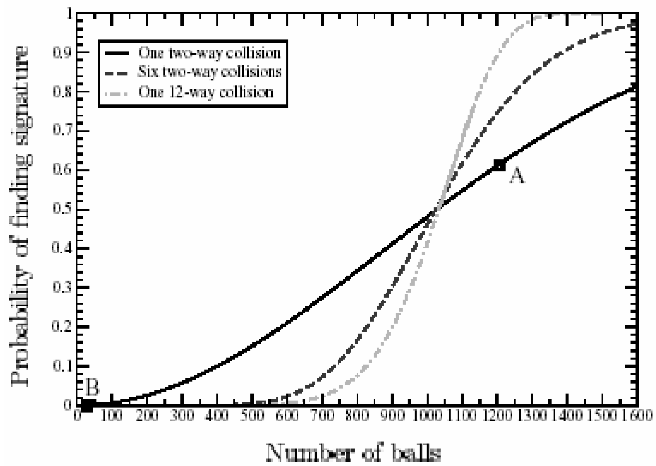
BiBa Signature Verification Algorithm

- Receiver obtains M and vector of SEALs.
- Receiver authenticates seals using the commitments previously obtained
- Receiver computes h= H(M).
- Assuming k=2, check s_i ?s_j, and $G_h(s_i) = G_h(s_j)$.
- Verification is computationally efficient

Security of the BiBa Signature

- Security comes from the difficulty of finding *k*-way collisions for one-way functions (similar to MicroMint).
- Exploits 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 hash of *k* random messages are distinct is:
    - $e^{-k(k-1)/2N}$, where N is range of hash function.

Security of The BiBa Signature



BiBa Security Considerations

- Upper bound on the probability that a adversary forge a signature:

  $P_f = \{(r\ k)\ (n-1)^{r-k}\ /\ n^{r-1}\}$
- Two main ways for attacker to attempt to forge signatures.
  - simply collect SEALs disclosed in signatures.
  - find SEALs by brute-force computation.
- Assumption is that attacker knows only a few SEALS compared to sender

BiBa Security Considerations (cont'd)

- Increasing k decreases probability (P_f) that attacker can find signature knowing k SEALs

k	n	P_f	k	n	P_f
2	762460	$2^{-19.5403}$	13	192	$2^{-91.0196}$
3	15616	$2^{-27.8615}$	14	168	$2^{-96.1001}$
4	3742	$2^{-35.6088}$	15	151	$2^{-101.3377}$
5	1690	$2^{-42.8912}$	16	136	$2^{-106.3119}$
6	994	$2^{-49.7855}$	17	123	$2^{-111.0802}$
7	672	$2^{-56.3539}$	18	112	$2^{-115.7250}$
8	494	$2^{-62.6386}$	19	104	$2^{-120.6079}$
9	384	$2^{-68.6797}$	20	96	$2^{-125.1143}$
10	310	$2^{-74.4851}$	21	89	$2^{-129.5147}$
11	260	$2^{-80.2237}$	22	83	$2^{-133.8758}$
12	222	$2^{-85.7386}$	23	78	$2^{-138.2788}$

Table 1: The security of some BiBa instances. The signer knows $t = 1024$ SEALs and the adversary has $r = k$ SEALs.

BiBa Signature Protocol Extensions for Increased Security

- Use multiple two-way collisions to generate a signature.
  - signature is composed of *z* pairs of SEALs.
- Multi-way collisions, instead of two-way collisions (i.e., k > 2).
- Use a multi-round scheme for generating the SEAL's

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 to forge signature.
- But this would limit the number of messages that can be signed.
- One solution: replenish the disclosed SEALs.
  - Use one-way hash chains similar to S-Key.

SEAL Chains

- Use two pseudorandom one-way functions (F and F')
- F is used to generate one-way SEAL chains and F' is used to generate chain of Salt values
- Generate chain of Salts recursively:
  - $K_i = F'_{Ki+1} (0)$ $(1 < i < l)$
- Use the Salt values to generate SEALs:
  - $S_{i,j} = F_{Si,j+1}(K_{j+1})$ $(1 < j < l)$

SEAL Chains

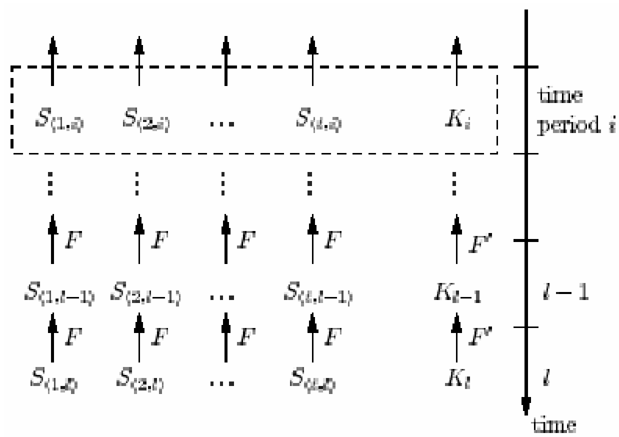


Figure 3: Using one-way chains to construct SEAL

NC STATE UNIVERSITY Computer Science

BiBa Broadcast Authentication Protocol

- Sender divides the time up into time periods of equal duration T_d.
- In each time period i, the SEALs $S_{x,i}$, and the salt K_i are *active*. *(1 <= x <= l)*
- As time advances an entire row of SEALs expires and a new row becomes active.

NC STATE UNIVERSITY Computer Science

BiBa Broadcast Authentication Protocol (cont'd)

- Sender publishes each salt at the beginning of the time period when it becomes active.
- Sender only discloses the active SEALs of a time period that are part of a BiBa signature.
- When a new receiver comes online, sender sends it all the SEALs and the salt of a previous time period over an authenticated channel (e.g., using RSA digital signature).

BiBa Broadcast Authentication Protocol (cont'd)

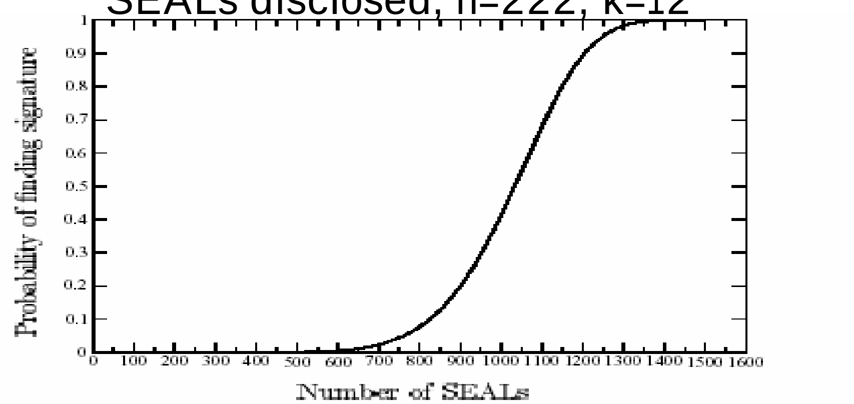
- Receiver authenticates salts by verifying K_i ? = $F'K_{i+1}$ (0).
- Receiver authenticates SEALs by following the one-way SEAL chain back to a SEAL that it knows is authentic.

BiBa Security Conditions

- Need to ensure that adversary knows few active balls
- Receiver can do this if it is time synchronized with sender
- Assume max time synch error *d* sec. between sender and receiver
- Sender cannot sign more than r/k messages in *d* sec. where r=max. # of SEALs the adversary can know and k=# of SEALs revealed in each signature
- If sender needs to send more than r/k messages in d sec. it needs to use multiple BiBa chains

BiBa Security Conditions (cont'd)

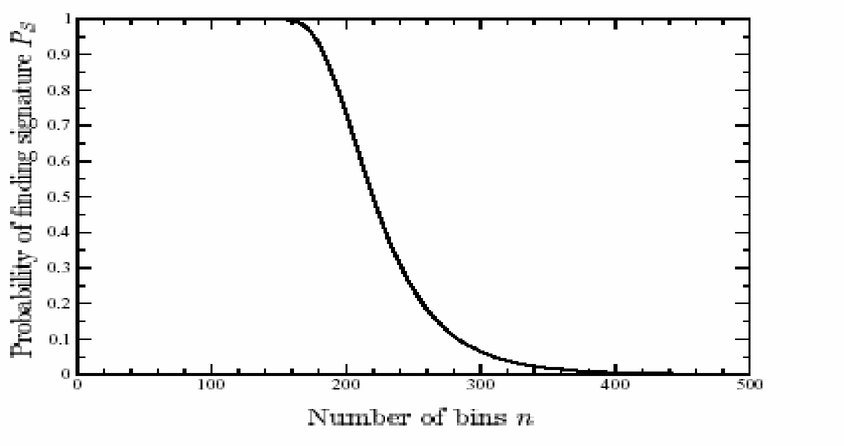
Note: X axis is number of SEALs disclosed, n=222, k=12



(b) $12 \leq x \leq 1600$

Selecting BiBa Parameters

- Note: k=12



BiBa Computational Requirements

	Computation	Memory
Precomputation	$l(t+1)T_F$	$l(m_1 + t \cdot m_2)$
Signature Generation	$(t \cdot T_G + T_H)/P_S$	$l(m_1 + t \cdot m_2)$
Signature Verification	$2 \cdot k \cdot T_G + T_H$	$m_1 + (k+n) \cdot m_2$

Table 2: BiBa Overhead. The salts are m_1 bits long, and the SEALs are m_2 bits long. The communication overhead (signature size) is $k \cdot m_2$ ($+m_1$ if we also send the salt).

BiBa Protocol Extensions

- BiBa has low communication overhead and robustnesses but still requires significant receiver computational overhead
- The base BiBa protocol has high receiver overhead because many of the generated SEALs are never used
- Develop two extensions to BiBa which provide tradeoffs between robustness and computational overhead
- The protocol extensions require every generated SEAL to be used
- Author refers to them as extensions “A” and “B”

BiBa Protocol Extension “A”

- Provides lower receiver overhead but no tolerance for packet loss
- The protocol extensions require every generated SEAL to be used
- Uses the concept of SEAL boundaries
- SEALs above the boundary are disclosed
- The sender and receiver always know the current boundary

BiBa Protocol Extension "A"

• In this case the SEAL boundary is (0,2,3,0,1,2)

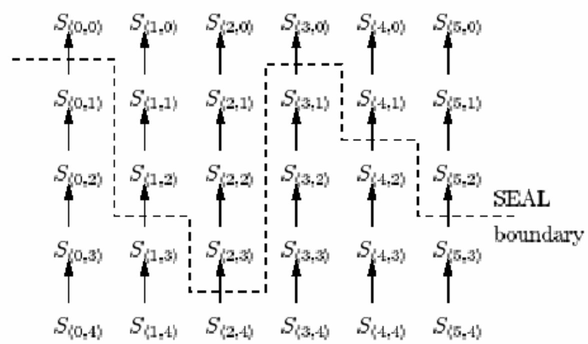


Figure 4: Using one-way chains to construct SEAL

NC STATE UNIVERSITY Computer Science

BiBa Protocol Extension "A"

- This scheme does not work if the attacker could slow down traffic delivery to receiver and collect a large number of SEALs below the boundary
- The attacker could then spoof the subsequent data traffic since it constantly receives fresh SEALs from sender
- Illustrates the need for time synchronization between sender and receiver so that the receiver knows the schedule for sending packets

NC STATE UNIVERSITY Computer Science

BiBa Protocol Extension “B”

- Provides tolerance for packet loss.
- Extension “A” does not tolerate packet loss because each receiver needs to know the SEAL boundary at all times
- Extension “B” includes the SEAL boundary in the information sent with each packet
- Two ways to accomplish this:
  - Absolute encoding- sends the index of each SEAL in the current boundary
  - Relative encoding- sends only the change from the previous boundary

NC STATE UNIVERSITY Computer Science

BiBa Protocol Extension “B” (cont’d)

- Extension protocol can tolerate “some” packet loss
- However, if there is a long period of packet loss, attacker could collect SEALs and forge subsequent packets by claiming a bogus boundary
- Receiver needs to receive at least one packet for every v= r/k packets (i.e., no more than v-1 consecutive lost packets)

NC STATE UNIVERSITY Computer Science

Efficient Public Key Distribution

- Sending the public key to all receivers can potentially be a bottleneck
- Can implement a more efficient method for sender but requires more time for receivers to boot-up
- Receivers collect SEALs while they receive signed messages and verify the salt chain
- Periodically sender broadcasts hash of all SEALs and Salt for one time period authenticated with traditional digital signature
- The receiver can authenticate signature and then use them to authenticate subsequent traffic
- Receiver needs to collect t*log(t) SEALs to ensure that it has one ball of each chain with high probability

Conclusions and Future Work

- BiBa makes use of the birthday paradox to construct a digital signature scheme using one-way functions without a trapdoor
- According to author, BiBa is the only broadcast authentication protocol to meet all design requirements
- Advantages of Biba over other approaches:
  - Smaller signature size
  - Smaller verification overhead
- Disadvantages of Biba
  - Larger public key
  - Higher signature generation overhead (can be parallelized)

Conclusions and Future Work

- Useful in settings where the signer can send the public key to the verifier efficiently, or where the verifier is constrained on computation power (e.g. PDA's).
- Potential for future work:
  - Attempt to parallelize the generation of signatures
  - Decrease the signature generation overhead (refer to "Better than Biba" paper)- may need to tradeoff on something else like public key size