WaterBear: Asynchronous BFT with Information-Theoretic Security and Quantum Security

Sisi Duan
Tsinghua University and Zhongguancun Laboratory

Haibin Zhang
Beijing Institute of Technology

Liehuang Zhu
Beijing Institute of Technology

Boxin Zhao
Zhongguancun Laboratory

Usenix Security 2023
Byzantine Fault Tolerance (BFT)

- Building block for blockchains

- Timing assumptions
  - Synchronous
    - Known upper bound for message transmission/processing
  - Partially Synchronous
    - Unknown upper bound
  - Asynchronous
    - No upper bound
Background

- **Computational security**
  - The adversary is restricted to probabilistic polynomial-time

- **Information-theoretic security**
  - The adversary is unbounded
  - Typically assuming secure or authenticated channels

- **Quantum security (no PKC)**
  - No public key cryptography (PKC)
Asynchronous BFT Paradigms

BKR

Ben-Or, Kemer, and Rabin (BKR)
PODC 1994

n-f ABA instances decide 1

HoneyBadger
CCS 2016

BEAT
CCS 2018

EPIC
DSN 2020

RedBelly
S&P 2021

*assumes partial synchrony

O(n³) message | O(Ln²+λn³logn) communication | information-theoretic | O(logn) time
--- | --- | --- | ---
O(n²) message | O(Ln²+λn³logn) communication | Not information-theoretic | O(1) time

CKPS

Cachin, Kusawe, Petzold, Shoup (CKPS)
CRYPTO 2001

MVBA

Dumbo
CCS 2020

Speeding Dumbo
NDSS 2022

*O(Ln²+λn²) communication can be achieved theoretically
BKR (PODC 1994) -> PACE (CCS 2022)

ABA becomes the bottleneck

BKR

\[
\begin{align*}
\text{RBC}_0 & \quad \text{ABA}_0 \\
\text{RBC}_1 & \quad \text{ABA}_1 \\
\text{RBC}_2 & \quad \text{ABA}_2 \\
\text{RBC}_3 & \quad \text{ABA}_3 \\
\text{RBC}_4 & \quad \text{ABA}_4 \\
\text{ABA}_0 & \\
\text{ABA}_1 & \\
\text{ABA}_2 & \\
\text{ABA}_3 & \\
\text{ABA}_4 & \\
\end{align*}
\]

n-f ABA instances decide 1

PACE (Zhang and Duan)

\[
\begin{align*}
\text{RBC}_0 & \quad \text{RABA}_0 \\
\text{RBC}_1 & \quad \text{RABA}_1 \\
\text{RBC}_2 & \quad \text{RABA}_2 \\
\text{RBC}_3 & \quad \text{RABA}_3 \\
\text{RBC}_4 & \quad \text{RABA}_4 \\
\text{RABA}_5 & \\
\text{RABA}_6 & \\
\end{align*}
\]

deliver n-f RBC instances

Significant performance gain compared to BKR

When f=30, the peak throughput of PACE-Pisa is 1.66x that of Dumbo, 3.6x that of BEAT (CCS 2018)

Fig. 5. Running time breakdown of Dumbo/2 and HoneyBadgerBFT on one random node.
A Closer Look at PACE Paradigm

- Challenges with RBC
  - Bracha’s broadcast (PODC 1984)
    - Information-theoretic
    - carry message payload in every step
    - $O(Ln^2)$ communication; not communication-efficient
    - WaterBear
  - CT RBC (SRDS 2015)
    - Quantum-secure
    - Uses hashes
    - $O(Ln+\kappa n^2logn)$ communication
    - WaterBear-QS
  - Can use recent advancement as well, e.g., EFBRB (PODC 2022), CCBRB (PODC 2022)
A Closer Look at PACE Paradigm

- **Challenges with ABA**
  - Most practical ABA rely on common coins
  - Instantiated with threshold signatures or threshold PRF

- **Our solution**
  - Use ABA with local coins
ABA from Local Coins

- The only known ABA from local coins
  - Bracha’s ABA (PODC 1984)
  - 3 phases of n parallel RBC instances
  - $O(n^3)$ message
  - $O(2^n)$ time complexity due to the use of local coins

- Our goals
  - Design more efficient local coin based ABA
  - Avoid querying coins as much as possible
  - Coin-free fast path

```c
01 Initialization
02 r ← 0 {round}
03 func propose(v)
04 iv_0 ← v
05 vset ← {0, 1} {valid binary values that will be accepted}
06 start round 0
07 round r
08 r-broadcast pre-vote_r(iv_r) {phase 1}
09 upon r-delivering n - f pre-vote_r(v) such that for each
   pre-vote_r(v), v ∈ vset {phase 2}
10 if there are n - f pre-vote_r(v)
11 decide v
12 iv_{r+1} ← v
13 vset ← {v}
14 else
15 v ← majority value in the set of pre-vote_r(v) messages
16 r-broadcast main-vote_r(v)
17 upon r-delivering n - f main-vote_r(v) such that for each
   main-vote_r(v), v ∈ vset {phase 3}
18 if there are at least n/2 main-vote_r(v)
19 vset ← {v}
20 else
21 v ← ⊥
22 vset ← {0, 1}
23 r-broadcast final-vote_r(v)
24 upon r-delivering n - f final-vote_r(v) such that for each
   final-vote_r(v), v ∈ vset for each final-vote_r(v), vset = {0, 1}
25 if there are at least 2f + 1 final-vote_r(v)
26 decide v
27 iv_{r+1} ← v
28 vset ← {v}
29 else if there are f + 1 final-vote_r(v)
30 iv_{r+1} ← v
31 vset ← {0, 1}
32 else
33 c ← Random() {obtain local coin}
34 iv_{r+1} ← c
35 vset ← {0, 1}
32 r ← r + 1
```

Figure 10: The Bracha’s ABA protocol [13]. The code for $p_r$.
Our Local Coin Based ABA

- Bracha’s ABA: $O(n^3)$ message, 9 steps in the fast path
- Cubic-ABA: $O(n^3)$ message, 5 steps in the fast path
- Quadratic-ABA: $O(n^2)$ message, 4 steps in the fast path

<table>
<thead>
<tr>
<th>ABA (local coins)</th>
<th>messages/round</th>
<th>steps/round</th>
</tr>
</thead>
<tbody>
<tr>
<td>Bracha’s ABA [14]</td>
<td>$n^3$</td>
<td>9 to 12</td>
</tr>
<tr>
<td>Cubic-ABA (this work)</td>
<td>$n^3$</td>
<td>5 to 7</td>
</tr>
<tr>
<td>Quadratic-ABA (this work)</td>
<td>$n^2$</td>
<td>4 or 5</td>
</tr>
</tbody>
</table>

Table 3: Local coin based ABA protocols with optimal resilience. We consider the messages and steps in each round. Messages/round and steps/round denote number of messages and steps among all replicas per round.
Our ABAs

- By replacing local coins with **weak common coins** or **comon coins**, we obtain more efficient ABA protocols compared to existing state-of-the-art ABA.

<table>
<thead>
<tr>
<th>ABA (weak common coins)</th>
<th>steps/round</th>
<th>rounds</th>
</tr>
</thead>
<tbody>
<tr>
<td>MMR15 [57, 2nd alg]</td>
<td>9 to 13</td>
<td>$d + 1$</td>
</tr>
<tr>
<td>Crain [26, 1st alg]</td>
<td>5 to 7</td>
<td>$d + 1$</td>
</tr>
<tr>
<td>CC-ABA (this work)</td>
<td>4 or 5</td>
<td>$d + 1$</td>
</tr>
</tbody>
</table>

Table 4: ABA protocols using weak common coins. Rounds denote the expected number of rounds. The total number of steps is a product of steps/round and rounds.

<table>
<thead>
<tr>
<th>ABA (common coins)</th>
<th>steps/round</th>
<th>rounds</th>
<th>good-case-coin-free</th>
</tr>
</thead>
<tbody>
<tr>
<td>MMR15 [57, 2nd alg]</td>
<td>9 to 13</td>
<td>3</td>
<td>yes</td>
</tr>
<tr>
<td>Cobalt [53]</td>
<td>3 or 4</td>
<td>4</td>
<td>no</td>
</tr>
<tr>
<td>Crain [26, 1st alg]</td>
<td>5 to 7</td>
<td>3</td>
<td>yes</td>
</tr>
<tr>
<td>Crain [26, 2nd alg]</td>
<td>2 or 3$^\dagger$</td>
<td>4</td>
<td>no</td>
</tr>
<tr>
<td>Pillar [64]</td>
<td>2 or 3</td>
<td>4</td>
<td>yes</td>
</tr>
<tr>
<td>CC-ABA (this work)</td>
<td>4 or 5</td>
<td>3</td>
<td>yes</td>
</tr>
</tbody>
</table>

Table 5: ABA protocols using perfect common coins. $^\dagger$The second algorithm of Crain relies high threshold common coins and is less efficient than Pillar. Compared to Pillar, CC-ABA has the good-case-coin-free property that is vital for the asynchronous distributed key generation protocol [30].
Our ABAs

**Cubic-ABA**

Idea: Use all-to-all communication to replace parallel RBC as much as possible

Bracha’s ABA involves 3 phases of n parallel RBC

**Quadratic-ABA**

Idea: Use all-to-all communication only

Any voted value needs to be ‘confirmed’ by counting the number of votes from the previous step
Local Coin Based RABA

- Bracha’s ABA
  - $O(n^3)$ message
  - 9 steps in the fast path

- Cubic-ABA
  - $O(n^3)$ message
  - 5 steps in the fast path

- Cubic-RABA
  - $O(n^3)$ message
  - 5 steps in the fast path

- Quadratic-ABA
  - $O(n^2)$ message
  - 4 steps in the fast path

- Quadratic-RABA
  - $O(n^2)$ message
  - 4 steps in the fast path

- RABA (CCS 2022)
  - if correct $f+1$ replicas propose 1, all correct replicas decide 1
  - Coin-free fast path
Our RABA

Quadratic-RABA

Idea: Use all-to-all communication only

Any voted value needs to be ‘confirmed’ by counting the number of votes from the previous step
Evaluation

- Golang
- Evaluated 5 protocols in total
  - 4 new ones (WaterBear family)
  - BEAT (CCS 2018)
- AWS m5.xlarge, 4 vCPU, 16GB memory
- up to 61 instances
Results

• All WaterBear-QS protocols outperform BEAT
  • \( n=16 \), WaterBear-QS-Q has 1/8 latency and 1.23x throughput compared to BEAT
  • Due to the use of PACE framework
• WaterBear-QS protocols consistently outperform WaterBear protocols
  • Communication is important!
• Building efficient quantum-secure asynchronous BFT is possible
WaterBear: Asynchronous BFT with Information-Theoretic Security and Quantum Security
Haibin Zhang, Sisi Duan, Boxin Zhao, Liehuang Zhu

- **Quadratic-ABA and Cubic-ABA**: Efficient local-coin based asynchronous binary agreement (ABA) protocols
- **WaterBear Family**: Efficient asynchronous Byzantine fault-tolerant (BFT) protocols with stronger security guarantees

Sisi Duan
Tsinghua University and Zhongguancun Laboratory
duansisi@tsinghua.edu.cn
Usenix Security 2023