Brahms

Byzantine-Resilient Random Membership Sampling
Bortnikov, Gurevich, Keidar, Kliot, and Shraer

Edward (Eddie) Bortnikov
Maxim (Max) Gurevich
Idit Keidar

Gabriel (Gabi) Kliot
Alexander (Alex) Shraer
Why Random Node Sampling

- Gossip partners
  - Random choices make gossip protocols work
- Unstructured overlay networks
  - E.g., among super-peers
  - Random links provide robustness, expansion
- Gathering statistics
  - Probe random nodes
- Choosing cache locations

The Setting

- Many nodes
  - 10,000s, 100,000s, 1,000,000s, …
- Come and go
  - Churn
- Every joining node knows some others
  - Connectivity
- Byzantine failures
  - f out of N
  - Standard model: message sources known
    - need robust ids
The MADness

- Byzantine nodes
  - Arbitrary behavior
  - Captures bugs, hacker intrusions, selfishness
- May want to bias samples
  - Eclipse (isolate) nodes, DoS nodes
  - Promote themselves, bias statistics
- To limit their power, use challenged messages
  - Require solving computational puzzle
  - Byzantine nodes can send portion $p$ of such messages

Previous Work

- Gossip membership
  - Small views - $O(\log N)$ [Lpbcast, Scamp, Cyclon, Alavena et al.]
  - Robust to churn and benign failures
  - Never proven uniform samples
  - Byzantine resilience needs full views [Fireflies, Drum, BAR]
- Random sampling in overlays, random walks
  [Saia, Massoulié et al., Gkantsidis et al., RaWMS]
  - Proven uniform
  - Needs overlay / topology with known connectivity
  - Not Byzantine resilient
- Byzantine-resilient (usually structured) overlays
  [Singh et al., Castro et al, Condie et al., Awerbuch&Scheideler]
  - Overcome eclipse attacks, secure routing
  - Overlays are just one application of sampling
Our Approach

1. Gossip-based membership
   - Logarithmic-size partial views
2. Limit the damage of Byzantine nodes
   - Using a bag of tricks
3. Precisely analyze how much of the view Byzantine nodes can still bias
   - Validate in simulations
4. Data Stream Sampling
   - Unbias the views
   - Converges to proven near-uniform samples

Brahms Components

- Gossiper – distributed
- DSSampler – local
**DSSampler – Data Stream Sampler**

- **Input:** data stream
  - N unique elements
  - Elements appear multiple times
  - Bias: some appear more than others
- **Output:** (nearly) uniformly random sample of unique element in stream
- **Space:** stores one value from the stream
- **Trick:** universal hashing
  - Choose hash value closest to a random point

**DSSampler Pseudo-Code**

```plaintext
init
  r ← random number in hash range
  h ← random hash function from H
  cur ← (0,0)
next (id)
  for k=1,…, K
    if h(id,k) closer to r than cur
      cur ← (id,k)
sample
  return cur.id
```

Node id has K virtual nodes: (id,1), …, (id, K)
Node id is sampled if r lies in the arc pertaining to any of id’s virtual nodes
Sample Properties

- Once all unique elements are observed, each sample is each node with probability $O(1/N)$
- Near-uniform
- Distance from uniform sample linear in portion of stream seen so far
  - Improves over time
- Consider a biased stream with $x\% > f$ bad ids
  - The average portion of bad ids in the sample is bounded by $x\%$ and goes to $f$ with time
- Resembles use of two routing tables in
  [Castro, Druschel, Ganesh, Rowstron, Wallach OSDI02; Condie, Kacholia, Sankararaman, Hellerstein, Maniatis NDSS06]
DSSampler Convergence

![Graph showing DSSampler Convergence](image)

Gossip-Based Membership: Primer

- Maintain a small local view
- View constantly changes
  - Essential due to churn
- Pull – probe a node in local view, get some ids from its view
  - “Mix” existing knowledge within the network
- Push – send my_id to others in my local view
  - Reinforce knowledge about nodes that are underrepresented in other nodes’ views
    - e.g., newborn nodes
Gossiper’s Bag of Tricks: Part 1

1. Control the portion of ids from push, $\alpha$, versus pull, $\beta$, in the local view
2. Use challenged messages for push
   - Together ensure an upper bound on the ratio of bad ids system-wide
     - See analysis below
   - Still, attacker can target a node
     - and isolate it
     - See analysis below
   - Can target all nodes one by one

Gossiper’s Bag of Tricks: Part 2

3. Detect attacks
   - Too many pushes arrive
     - Due to randomness, use conservative threshold
   - Block view changes under attack
4. Reinforce view with history samples
   - Portion $\gamma$ of local view
taken from unbiased samples produced by DSSampler
   - Analyze views without this trick
   - Takes time to help
Gossiper Rounds

Old Local View

send push (challenged)

send pull request

respond to push challenges, pull requests
collect challenged pushes, pull responses

pushed ids (challenged)
pulled ids (requested)

New Local View

unbiased sample

Analysis 1: Portion of Red Ids
(Local View before Un-Biasing)

- A red local view entry is an entry containing an id controlled by the attacker
  - Could be either faulty or correct node id
  - We don’t know the attackers goals
- Ignore history samples for now
  - They only help
  - $\gamma = 0$
- Let $x(t)$ = portion of red ids in correct node views at time $t$
- Compute $E[x(t+1)]$ as function of $x(t)$, $p$, $\alpha$, $\beta$, $\gamma$
Portion of Red Ids: Impact of Push

Time t:
- Lost push
- Push

Time t+1:
- Push from faulty node

Portion of red pushes to correct views \( \phi(x(t)) \):
\[
\phi(x(t)) = \frac{p}{p + (1 - p)(1 - x(t))}
\]

Portion of Red Ids: Impact of Pull

Time t:
- Pull from faulty
- Pull from i: red with probability \( x(t) \) (simplified)

Time t+1:

\[
\phi(x) = \frac{p}{p + (1 - p)(1 - x)}
\]

\[
E[x(t+1)] = \alpha \phi(x(t)) + \beta (x(t) + (1-x(t)) \cdot x(t))
\]
Analysis: Portion of Red Ids (Cont’d)
(Local View before Un-Biasing)

In the paper, we:

1. Find fixed points $t_0$ of $x(t)$
   - Where $E[x(t_{0}+1)] = x(t_{0})$
   - As a function of $p$, $\alpha$, $\beta$, $\gamma$
   - For $\alpha=\beta=0.5$, $p < 1/3$, exists fixed point < 1
     - i.e., not all the view is poisoned

2. Show convergence to the above fixed point
   - From any initial portion < 1 of faulty ids
   - From [Hillam Theorem 1975]

3. Validate this using simulations
   - Start from various initial portions of faulty ids

4. Show that under uniform attack, every node’s portion of red ids converges to global fixed point

Portion of Red Ids in Fixed Point
(Local View before Un-Biasing)

With a few history samples, any portion of bad nodes can be tolerated

Assumed perfect in analysis, real history in simulations

Perfectly validated fixed points and convergence
Convergence to Fixed Point
(No History Samples)

Simulation:
Local view

Analysis:
$x(t)$

Analysis 2: Isolating Nodes

- With no attacks, time to partition grows exponentially in view size [Alavena et al.]
- Q: How fast can an attacker targeting a node cause that node to partition from the rest?
- Without history samples $\gamma = 0$
  - Theorem: Sub-linear time in local view size
  - Attacker can isolate nodes one by one
- With history samples
  - Exponential time in view size to isolate “veteran” nodes
    - If appeared in enough views or saw enough ids
  - Self-healing from such (rare) temporary partitions
  - One by one isolation impossible
Time to Isolate Targeted Node – No History Samples

Bigger views do not help much...

History starts helping before partition occurs!
Note: attack begins when all samples are empty.
History Samples: Rationale

- Judicious use essential
  - Bootstrap, avoid slow convergence
  - Deal with churn
- With a little bit of history samples (10%) we can cope with any \( p < 1 \)
  - Analysis assumes history is perfect
    - E.g., first \( p < 1/3 \), then attacker takes over more nodes
  - Amplification!
- One by one isolation impossible
  - By the time targeted node is isolated, others have histories to work with
  - Self-healing

Summary: Main Features

- Log-size views
- Resist Byzantine failures of linear portion
- Convergence to proven uniform samples
- Precise analysis of impact of failures