Introduction & Motivation

**Context:** Distributed, in-network aggregation
- Network monitoring, sensor/net/p2p query processing, ...
- Data is distributed – cannot afford to warehouse!
- Approximations are often sufficient
  - Can tradeoff approximation quality with communication

**Querier:** “How many Win-XP hosts running patch X have CPU utilization > 95%?”

**“Predicate poll” query**

More general aggregate queries (SUM, AVG), general-purpose summaries (e.g., random samples) of (sub)populations
In-Network Aggregation

Typical assumption: Benign aggregation infrastructure
- Aggregator nodes cannot “misbehave”

BUT, aggregators are often untrusted!
- 3rd party hosted operations (e.g., Akamai), shared infrastructure, viruses/worms, ...

Challenge: Verifiable, efficient, in-network aggregation
- Provide trustworthy, guaranteed-quality results with potentially malicious aggregators

Our Contributions

Proof Sketches: Family of certificates for verifiable, approximate, in-network aggregation
- Concise sketch synopses ➔ Communication-efficient
- Guarantee detection of malicious tampering w/hp if result is perturbed by more than a small error bound

Basic Technique: Combines FM sketch with compact Authentication Manifest (AM)
- Prevents inflation through crypto signatures; bounds deflation through complementary deflation detection

Extensions: Verifiable random sampling; verifiable aggregates over multi-tuple nodes
**Talk Outline**

- *Introduction and Motivation*
- *Overview of Contributions*
- System Model
- AM-FM Proof Sketches
- Extensions
  - Verifiable random samples
  - Verifiable aggregation over multi-tuple nodes
- Experimental Results
- Conclusions

---

**System Model**

\[ U = \text{size of sensor population} \]

*Inflation Attacks:* Aggregators can manipulate or inject spurious PSRs

*Deflation Attacks:* Aggregators can suppress valid PSRs
A Naïve Inflation Detector

*Straightforward application of crypto signatures*
- Each sensor node crypto signs each tuple satisfying the predicate poll, and sends up the tuple + signature
- Aggregators simply union the signed tuple sets and forward up the tree

Aggregators cannot forge sensor tuples
- Within crypto function guarantees

BUT, size of Authentication Manifest (AM) = size of answer set
- $O(U)$ in general!

Solution: AM-FM Proof Sketches

Sketch and AM structure of size only $O(\log U)$

Based on the *FM sketch* for distinct-element counting

**Bitmap of size** $O(\log U)$

<table>
<thead>
<tr>
<th>0</th>
<th>1</th>
<th>2</th>
<th>3</th>
<th>...</th>
<th>$k$</th>
</tr>
</thead>
</table>

$h()$ with prob $1/2^{k+1}$

$P[h(x)=0] = 1/2$, $P[h(x)=1] = 1/4$, $P[h(x)=2] = 1/8$, ...

Index of rightmost zero $\sim \log(\text{Count})$

$O(\log(1/\delta)/\varepsilon^2)$ sketches to get an $(\varepsilon, \delta)$–estimate of the Count
**Adding AM to FM: Inflation Prevention**

*Observation:* Each FM sketch bit is an independent function of the input tuples

AM = Authenticate each 1-bit in the FM sketch using a *signed "witness"/exemplar sensor tuple*

- Crypto-signed tuple that turns that bit on

**Aggregators:** Merge input PSRs (AM-FM sketches)

- **OR** the FM sketches
- Keep a single exemplar for each 1-bit

```
<13,...>  <t1,a1,s(a1,t1)>
1 1
<14,...>  <t2,a2,s(a2,t2)>
1 1
```

- **Size** = $O(\log U)$ – Cannot forge 1-bits

**AM-FM Proof Sketches: Bounding Deflation**

Malicious aggregator can omit 1-bits & witnesses from sketch $\rightarrow$ Underestimate predicate poll count

**Approach:** *Complementary Deflation Detection*

- Assumes that we know sensor count $U$
- Use AM-FM to estimate count for both $pred$ and $!pred$
- Check that $C_{pred} + C_{!pred}$ is close to $U$ (based on sketching approximation guarantees)
  - Adversary cannot inflate $C_{!pred}$ to compensate for deflating $C_{pred}$
  - Sum check will catch significant deviations
More Formally...

Assume $O(\log(2/\delta)/\epsilon^3)$ AM-FM proof sketches to estimate $C_{\text{pred}}$ and $C_{\text{lpred}}$

**Verification Condition:** Flag adversarial attack if $C_{\text{pred}} + C_{\text{lpred}} < (1-\epsilon)U$

**Theorem:** If verification step is successful, the AM-FM estimate is within $\pm 2\epsilon U$ of the true $C_{\text{pred}}$ whp
- Adversary cannot deflate the result by more than $2\epsilon U$ without being detected whp
- Relative error guarantees for high-selectivity predicates

Verifiable Random Sampling

Build a general-purpose, verifiable synopsis of node data
- Can support arbitrary predicates, quantile/heavy-hitter queries, ...

Traditional (eg, reservoir) sampling + authentication fails
- Adversary can arbitrarily bias the sample

**Solution:** AM-Sample Proof Sketches
- Use FM hashing to sample, retain tuples + AMs for all tuples mapping above a certain level
- *A la* Distinct Sampling [Gibbons’01] – adapt level based on target sample size
- Easily merged up the tree using max-level
- Verification condition and error guarantees based on target sample size and knowledge of U
Aggregates over Multi-Tuple Nodes

So far, focus on predicate poll queries

- Each sensor contributes $\leq 1$ tuple to result

**Key Issue:** Knowing the total number of tuples $M$

- With known $M$, our earlier results and analysis apply

**Approach:** Verifiable approximate counting algorithm

- Estimate $M$ using a logarithmic number of simple AM-FM predicate polls
- To within a given accuracy $\theta$, using predicate polls of the form

\[ \text{Fraction of sensors with } \#\text{tuples} \geq (1+\theta)^k \]

- Detailed algorithm, analysis, ... in the paper

Other Extensions / Issues

Discuss “generalized template” for proof sketches to support verifiable query results

- E.g., Bloom-filter proof sketch

**Accountability:** Trace-back mechanisms for pinpointing attackers

Only approximate knowledge of population size $U$
Experimental Study

Study *average-case behavior* of AM-FM proof sketches for verifiable predicate polls

Population of 100K sensors, fixed number of sketches to 256
- About 4% of space for “naïve”
- \( \varepsilon \approx 0.15 \) wp 0.8

Parameters
- Predicate selectivity
- “Coverage” of malicious aggregators
- Two adversarial strategies (Targeted, Safe)

Some Results: Benign Population

![Graphs showing the results for benign population.](image)

- False positives and false negatives as functions of predicate selectivity.
- Alarms and out-of-bounds values for different selectivity levels.
**Experimental Summary**

Average case behavior is better than (worst-case) bounds suggest

- Adversary has even less “wiggle room” to deflate result without being detected

Bounds based on worst case for sketch approximation and combination of $\text{pred}/!\text{pred}$ estimates

Adversary typically has limited coverage in the aggregation tree

- Can only affect a small fraction of the aggregated results

---

**Conclusions**

Introduced *Proof-Sketches* – *first* compact certificate structure for verifiable, in-network aggregation

Basic technique: AM-FM proof sketch

- Adds concise AM to basic FM sketch; prevents deflation through *complementary deflation detection*

*Extensions*

- Verifiable random sampling
- Approximate verifiable counting for general aggregates over multi-tuple nodes

*Future:* Extending ideas and methodology to more general approximate in-network queries (e.g., joins)
Thank you!

http://www.cs.berkeley.edu/~minos/
minos@yahoo-inc.com,  minos@cs.berkeley.edu

Some Results: Safe Adversary