An Alternative to Fixed-Key Based Pre-Indexing

Joel C. Frank • Shayna M. Frank • Thomas M. Kroeger
Ethan L. Miller • Darrell D. E. Long

Center for Research in Storage Systems
University of California, Santa Cruz

Storage Systems Research Center
Center for Research in Storage Systems
Baskin Engineering UC Santa Cruz

The Fixed Key Dilemma

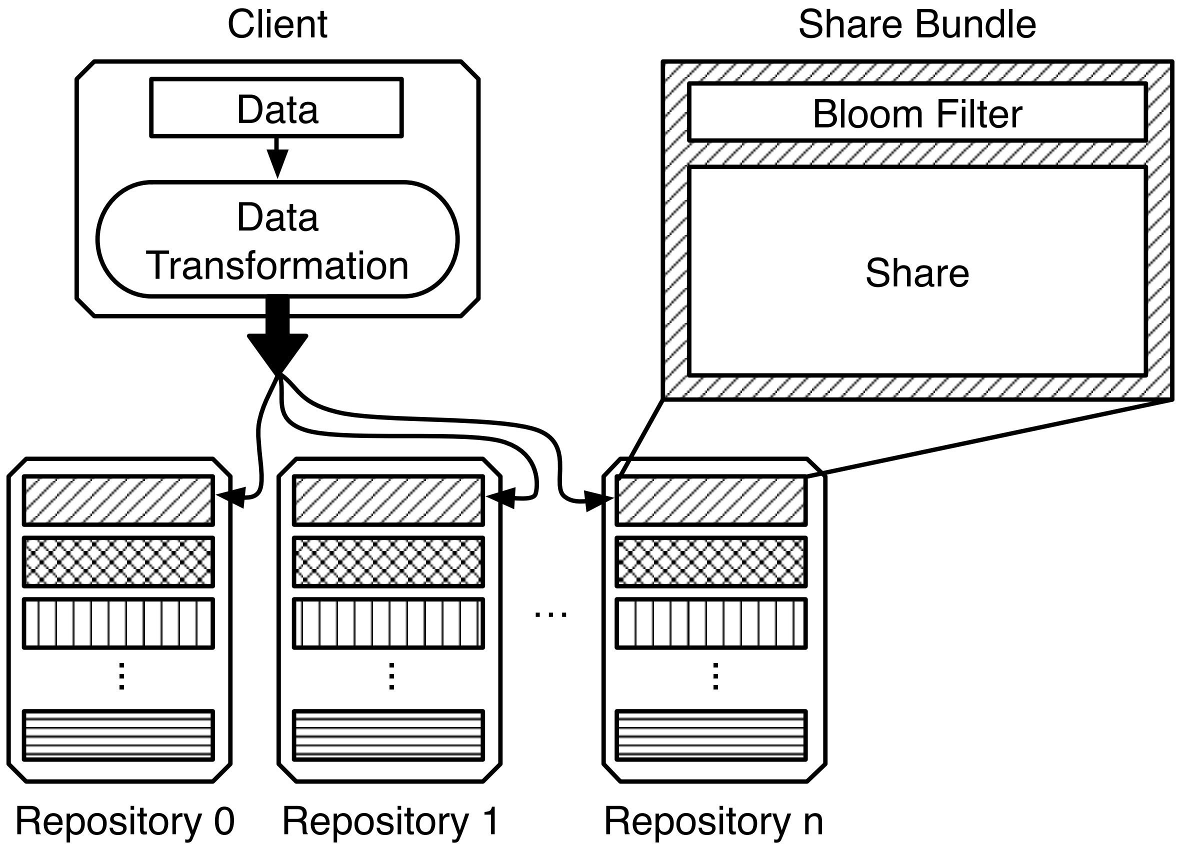
- Secret splitting (POTSHARDS)
  - Divide each data object into multiple “shares”
    - Any “sufficiently large” subset of shares can be used to recover the original object: number of shares and threshold can be customized
    - Fewer shares reveals ***no*** information
    - Minimizes insider threat: information-theoretic secure data protection
  - Independent sites: no single point of failure or compromise
  - System can operate in the face of single-site adversaries
- But without pre-indexing, searching is…
  - Unavailable, or
  - Requires data reassembly: reintroduces single point of failure or compromise
- Current pre-indexing methods rely on fixed-key encryption
  - Introduces single point of compromise
  - Not well suited for archival storage

Overview

- Goal: enable search without the need for reassembly

- Solution: Tag shares using Bloom filters containing search terms
  - Terms are inserted into the filters using salted hashing
  - Perform blinded searching of secret split data store
  - Known quantity of information release

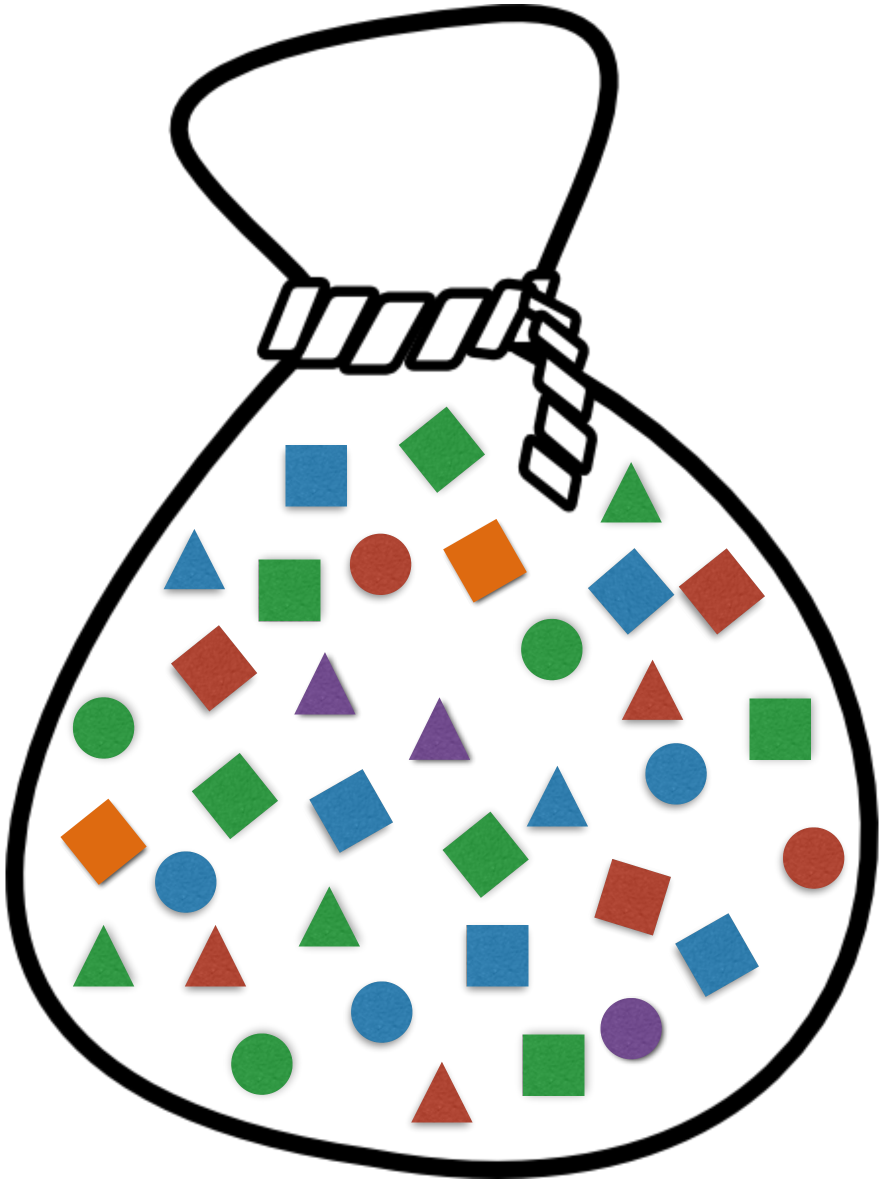
- Resulting system
  - Secure and searchable data store
  - Aids in information sharing
  - Assumes insider threat
    - Single repository
    - No collusion between attackers



What’s a Bloom filter?

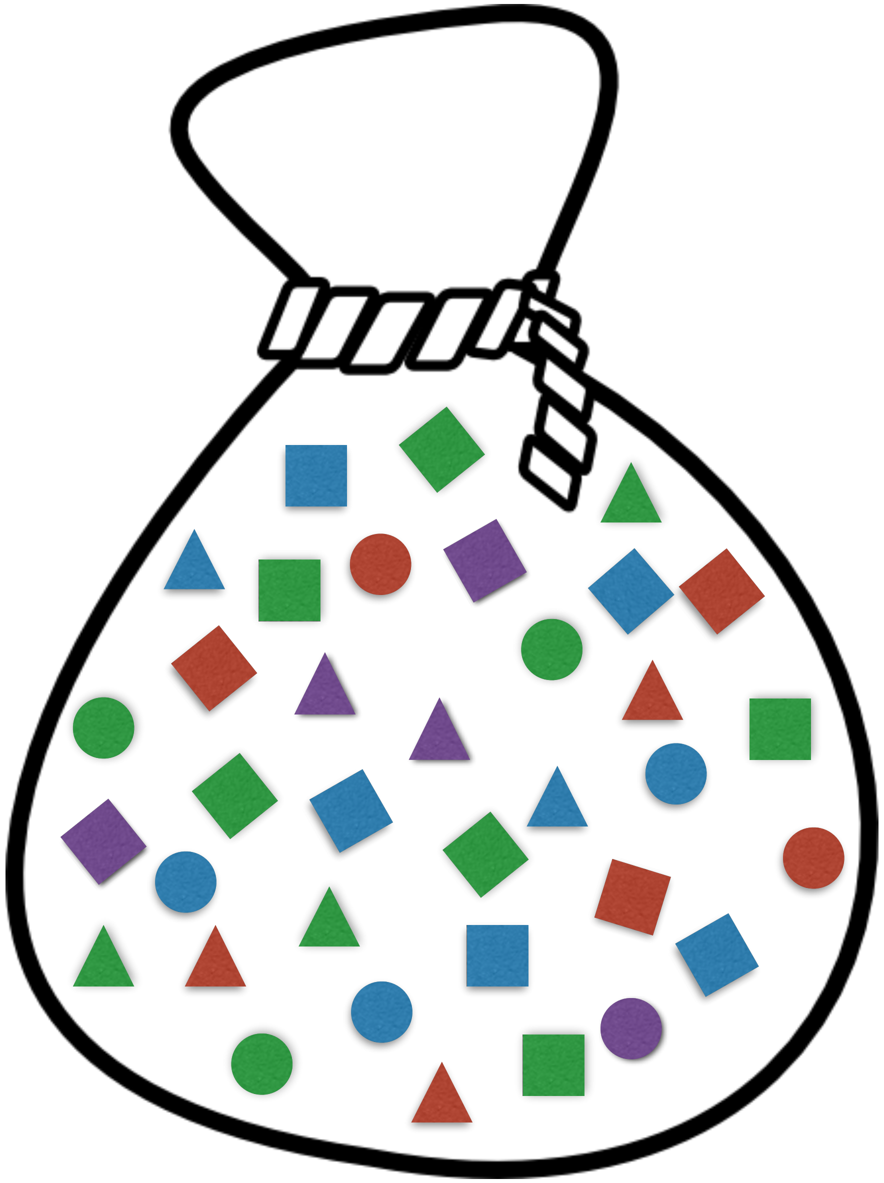
A way to store (approximate) answers to questions

- Given: A bag of different colored shapes
- Store questions and answers beforehand:
  - Blue shapes : yes
  - Circles : yes
  - Yellow shapes : no
  - etc…
- Queries:
  - Any purple triangles? Yes!
  - Any yellow circles? No!
  - Any purple squares? **Yes**!
    - We have both purple shapes and squares



Blinded searching

- How can we hide the properties of the data set?
  - Shrink the number of stored questions?
  - Reduce the number of properties?
  - Add “fake” properties?
- How can we make queries less useful to an adversary?
  - Ask for things we don’t really want?
- Together, these changes:
  - Decrease the uniqueness of the result set
  - Confuse the bag holder: more difficult to gather information
- But they make searching more difficult
  - Result set has more “useless” answers
  - Can user easily filter them out?



Ongoing work

- Currently testing system using digital corpora
- Quantify information released
  - Ensure that this approach doesn’t release useful information to an attacker
- Improving reconstruction performance
  - Query on each archive returns a set of shares from different documents
    - Shares from “desirable” should be in all result sets
    - But there might be many other shares…
  - Reduce the penalty due to “false hits”: identify the “undesirable” shares
  - Drastically reduce data reconstruction time
- Improve query performance by:
  - Organizing shares on each repository
  - Bloom filter variants

Questions?

Thank you!

Ethan L. Miller
elm@ucsc.edu

Joel C. Frank
jcfrank@soe.ucsc.edu