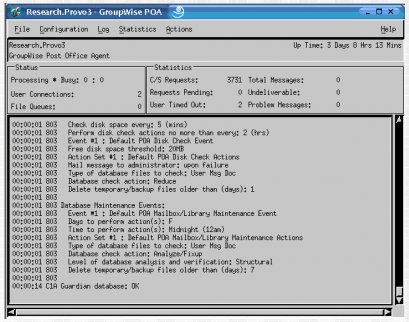
Secure Anonymous Database Search

Mariana Raykova
Binh Vo
Steven Bellovin
Tal Malkin


Memorial Hospital
EMERGENCY

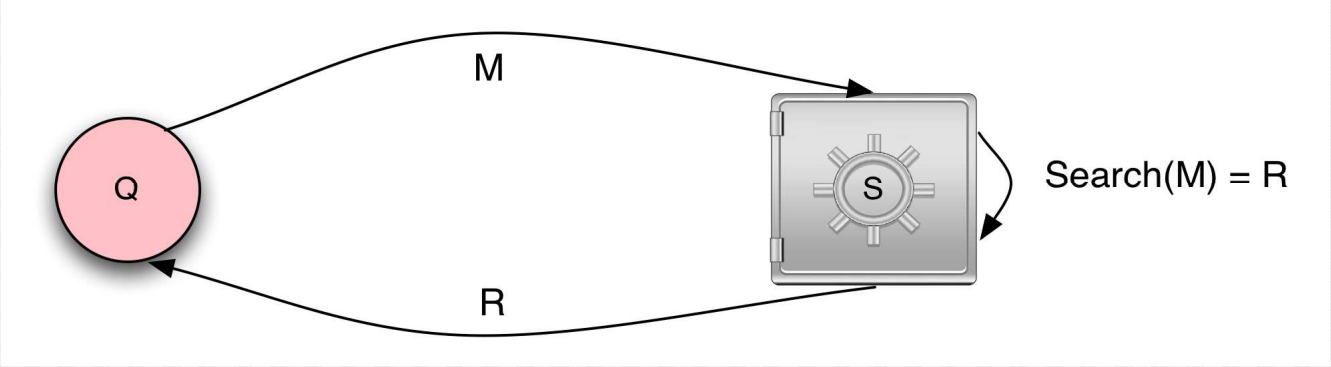

DEPARTMENT OF JUSTICE
FEDERAL BUREAU OF INVESTIGATION


CENTRAL INTELLIGENCE AGENCY


Research.Provo3 - GroupWise POA
File Configuration Log Statistics Actions Help
Research.Provo3
GroupWise Post Office Agent
Up Time: 3 Days 8 Hrs 13 Mins
Status
Processing * Busy: 0 : 0
User Connections: 2
File Queues: 0
Statistics
C/S Requests: 3731
Requests Pending: 0
User Timed Out: 2
Total Messages: 0
Undeliverable: 0
Problem Messages: 0
00:00:01 803 Check disk space every: 5 (mins)
00:00:01 803 Perform disk check actions no more than every: 2 (hrs)
00:00:01 803 Event #1 : Default POA Disk Check Event
00:00:01 803 Free disk space threshold: 20MB
00:00:01 803 Action Set #1 : Default POA Disk Check Actions
00:00:01 803 Mail message to administrator; upon failure
00:00:01 803 Type of database files to check: User Msg Doc
00:00:01 803 Database check action: Reduce
00:00:01 803 Delete temporary/backup files older than (days): 1
00:00:01 803
00:00:01 803 Database Maintenance Events:
00:00:01 803 Event #1 : Default POA Mailbox/Library Maintenance Event
00:00:01 803 Days to perform action(s): F
00:00:01 803 Time to perform action(s): Midnight (12am)
00:00:01 803 Action Set #1 : Default POA Mailbox/Library Maintenance Actions
00:00:01 803 Type of database Files to check: User Msg Doc
00:00:01 803 Database check action: Analyze/Fixup
00:00:01 803 Level of database analysis and verification: Structural
00:00:01 803 Delete temporary/backup Files older than (days): 7
00:00:01 803
00:00:14 CIA Guardian database: OK

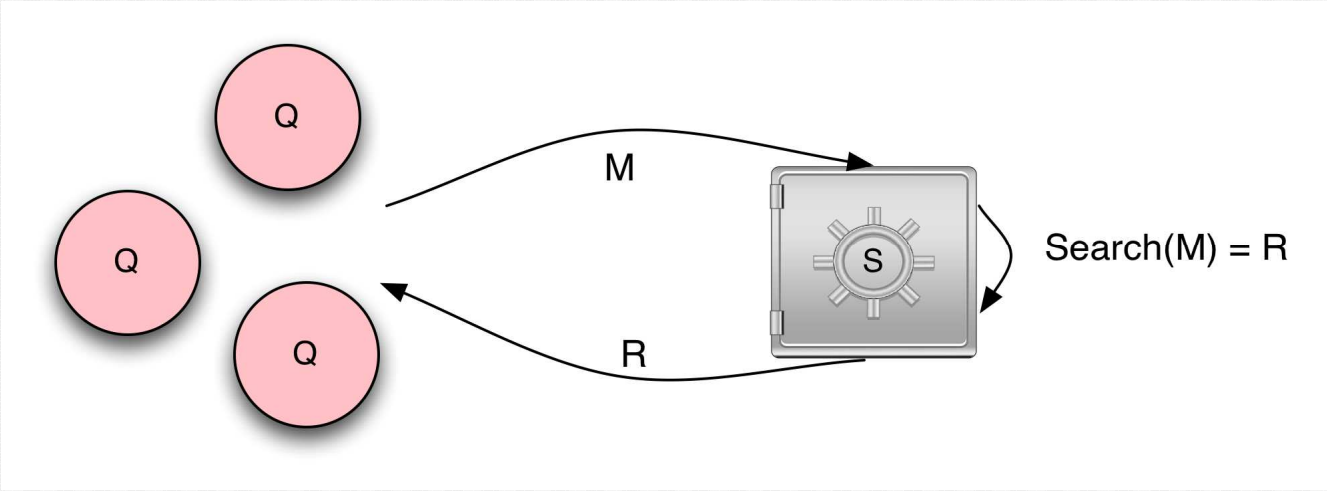
The underlying problem

- Goal: Controlled data sharing
- When protecting content, how do parties know if they have *data worth sharing*?
- Anonymous search



Further system requirements

- Search efficiency - sublinear
- Multiple parties
  - authentication – limit parties that can search
  - anonymization - hide querier identity



Our solution

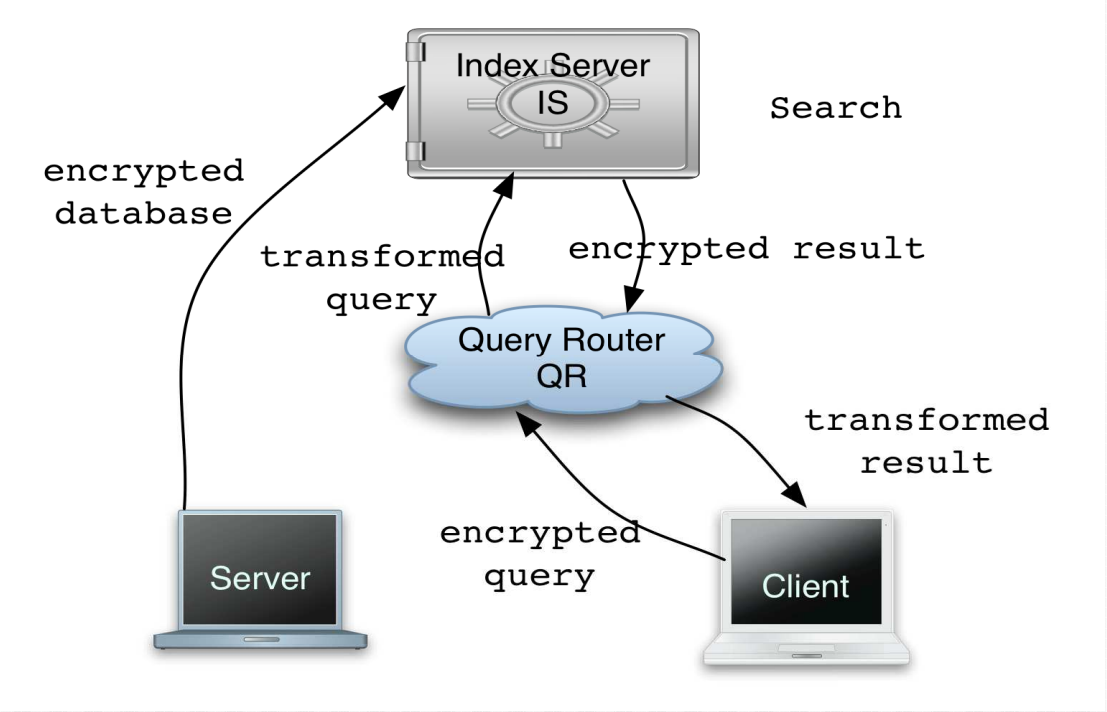
- System architecture
- Building blocks
- Analysis
- Implementation
- Test results

Search

- What is efficient search? – sublinearity
  - *decryption capability for matching ciphertext* does not work
- How to achieve?
  - deterministic encryption [BBO07] – *high min entropy of plaintext domain, replace randomness with hash*
  - Bloom filters
- Trade-offs
  - relaxed security notions – *equality pattern leaked*
  - false positives – *can be bounded*

System architecture

- Index Server – encrypted search
- Query Router – authentication and user anonymity



Re-routable encryption

- Goal
  - A has some information
  - A trusts B to distribute, but not to see
  - How to control distribution?
- Ciphertext transformation under different keys
  - Encryption scheme with group property

PH-DSAEP+

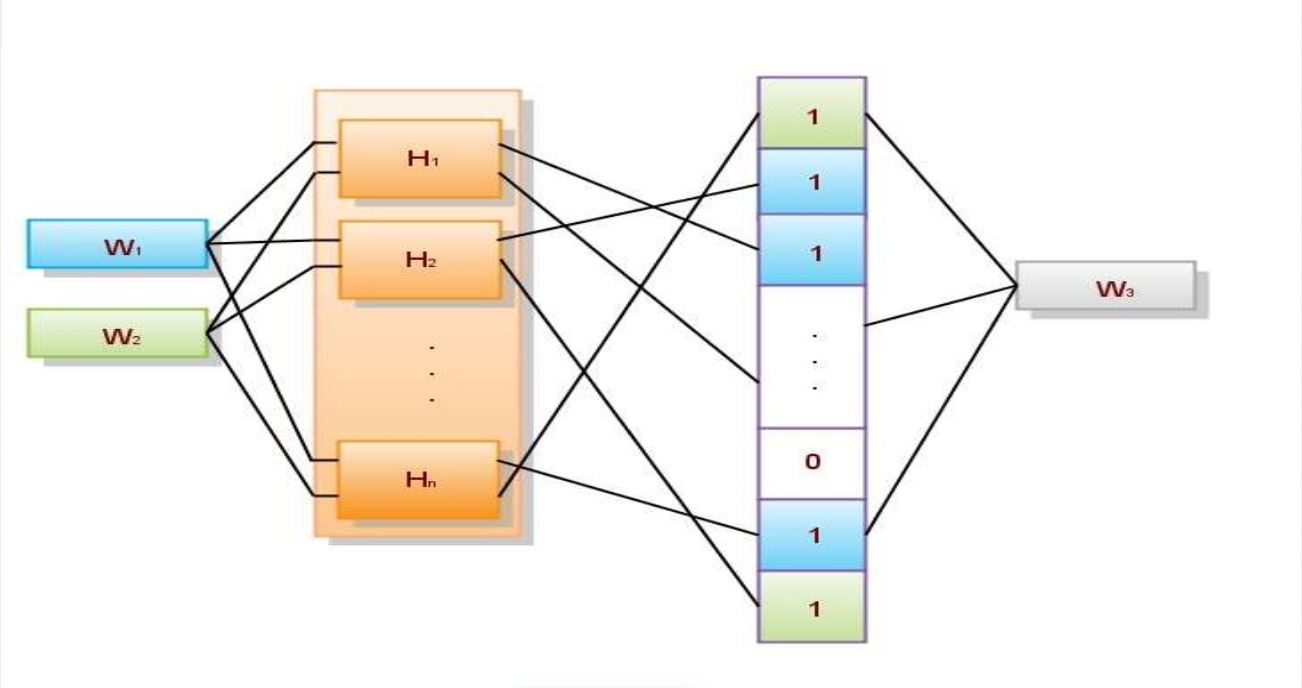
- Private key deterministic encryption – following BBO07
- Pohlig-Hellman function
  - *Group property*:

$$PH_{k1}(PH_{k2} (x)) = PH_{k1*k2}(x)$$

- Message padding **SAEP+** [Boneh01]
  - Randomness r replaced by a hash

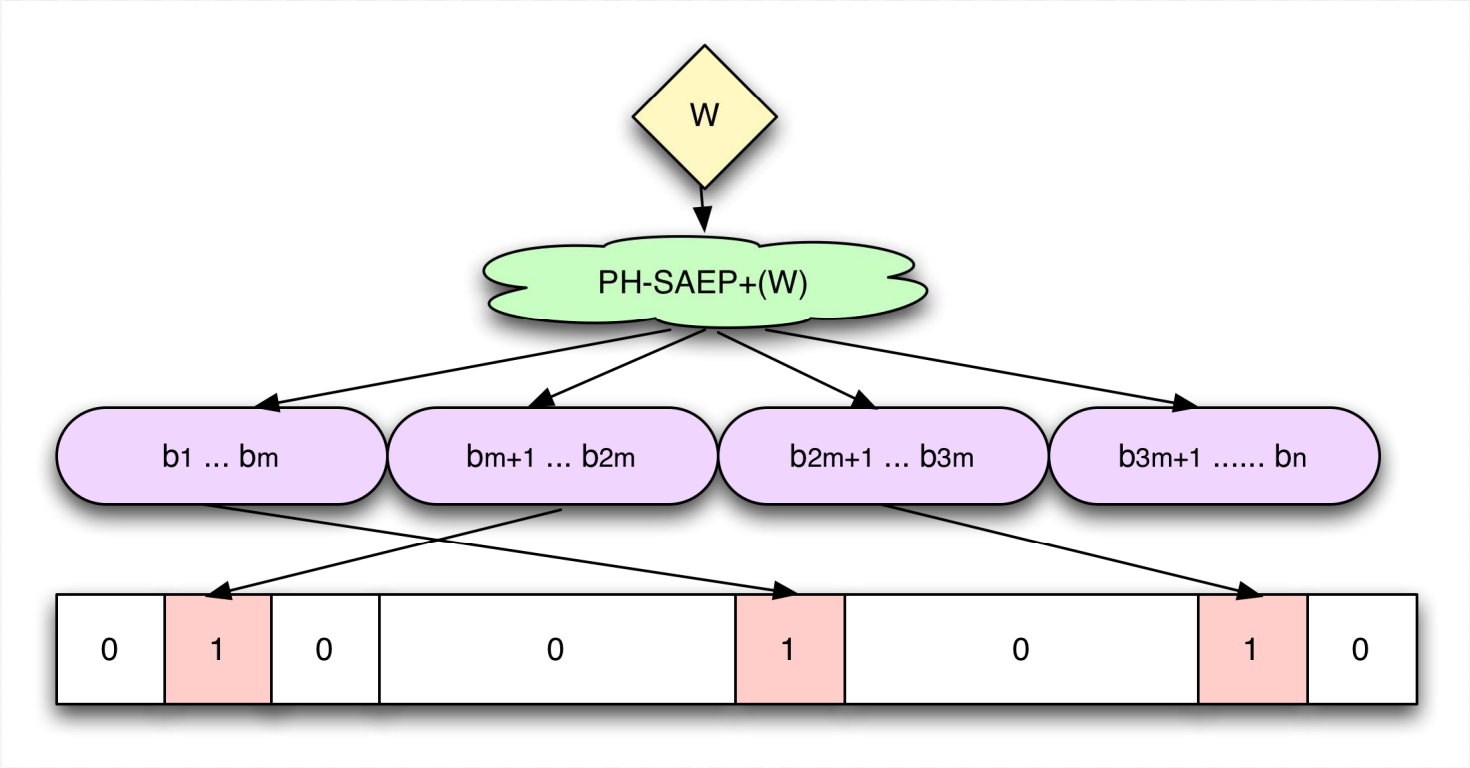
Bloom Filter Efficient Search

- Bloom filters – extend the idea of hashing

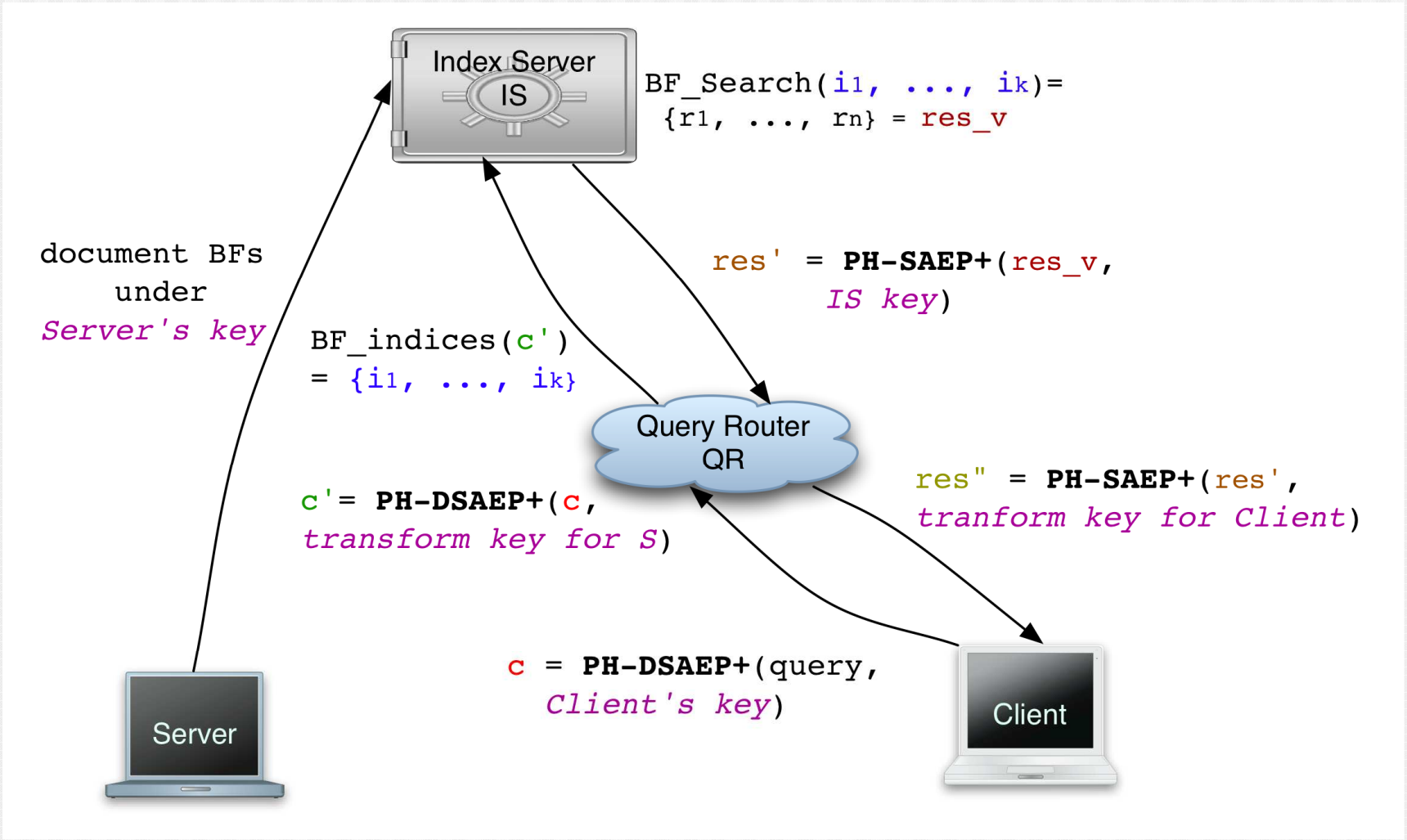


BFs for Document Search

- BF per document with stemmed words entries



Secure Anonymous Database Search (SADS)



Trust Assumptions – IS, QR

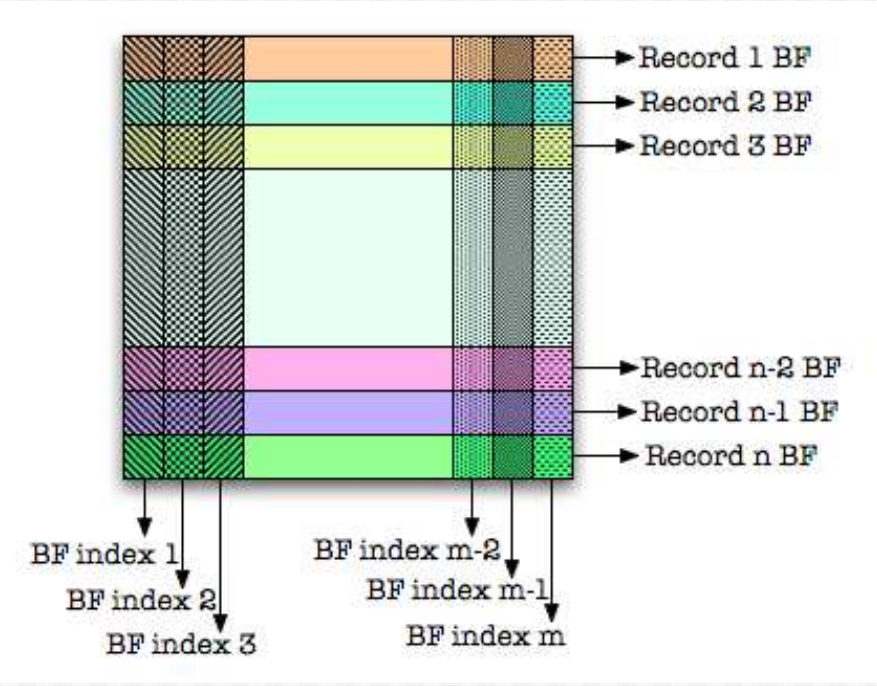
- Trust distribution – semi-honest IS, QR:
  - QR - *correct key transformation*
  - IS - *correct BF search*
- Privacy with respect to IS
  - IS *does not know relation of BFs to documents*
  - Client anonymity - *cannot link queries of one client*
- Privacy with respect to QR
  - Query privacy – *up to equality*, PH-DSAEP+
  - Result privacy

Security Guarantees

- Server participates only in preprocessing
- Client
  - Authenticated by QR
  - Learns only relevant result – *adjustable false positive rate, no false negatives*
- Collusion of IS and QR:
  - *Search pattern* in results leaked
  - *No search capability* - cannot submit queries

Index implementation

- What is bitslicing?
  - View a set of BFs as a matrix
  - Transpose
  - Track 'zeroed' slices
- What is gained?
  - Don’t read unnecessary
  - Cache behavior



Better Boolean queries

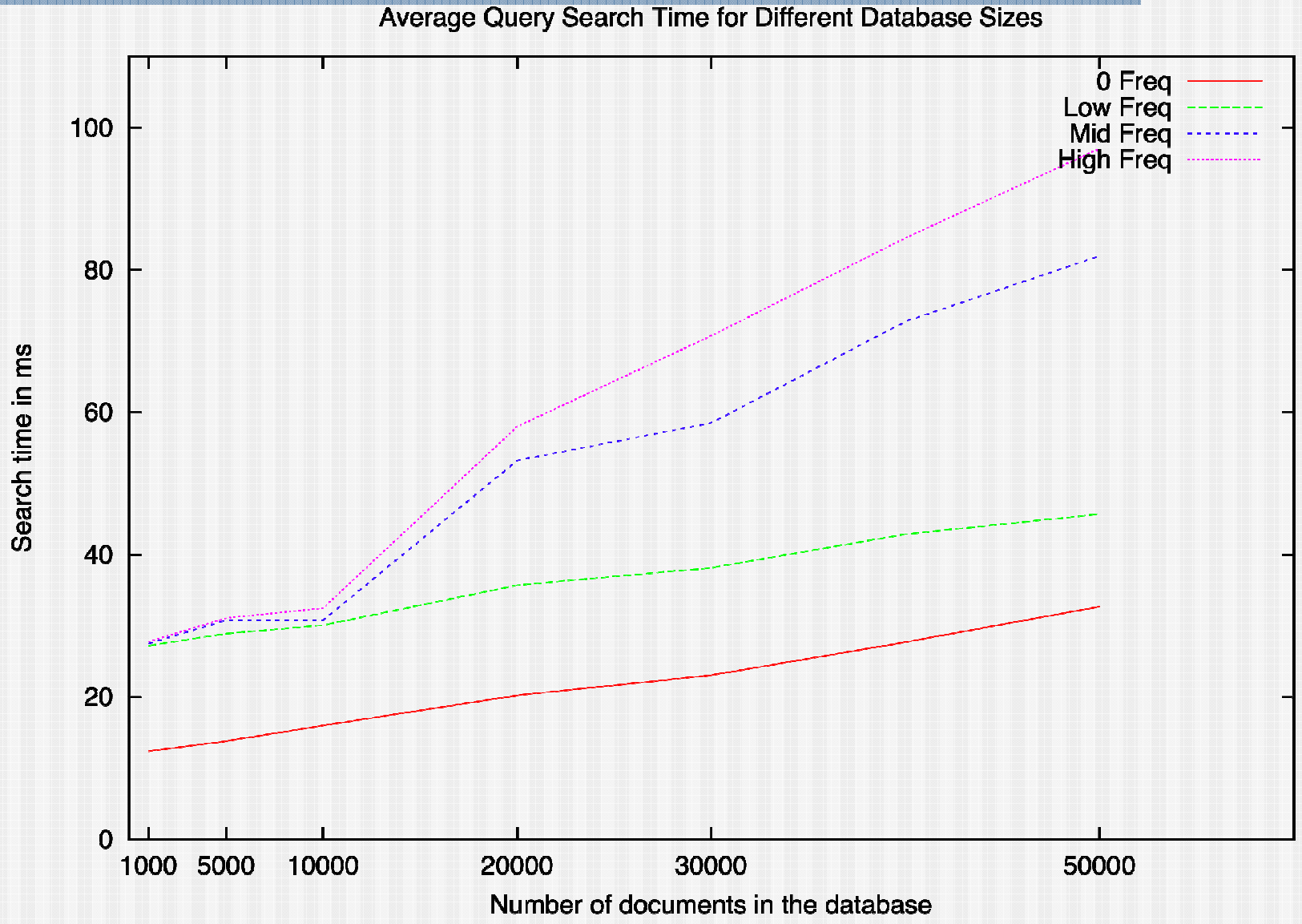
- The naïve way to do and/or queries
  - Run term queries in parallel
  - Union/intersect
- How we can do it better in sliced indexes
  - AND queries unioned in query indices
  - OR queries processed in parallel
  - OR query indices are handled in order of frequency in queries

Performance

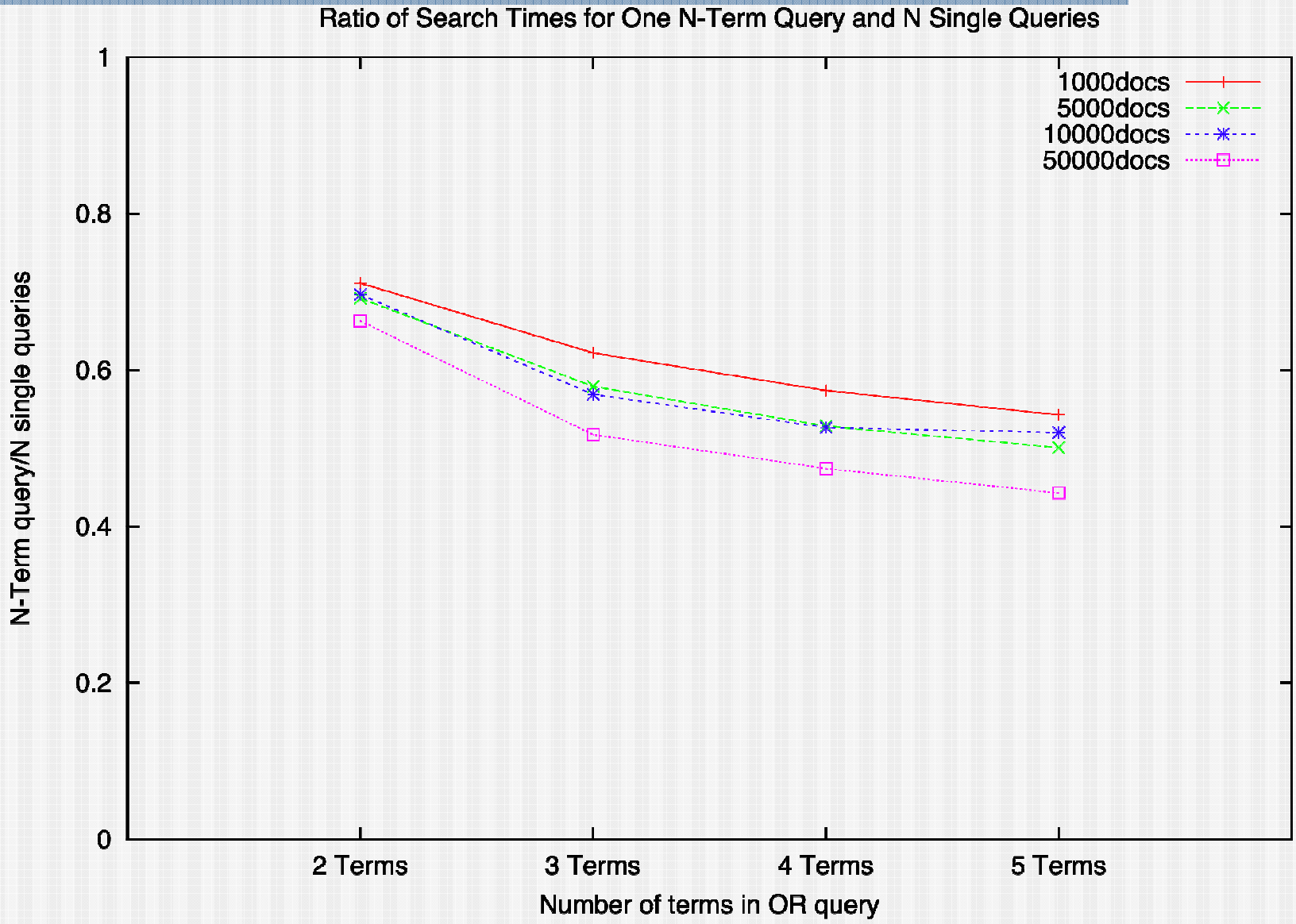
- *Constant search* time per BF
- *Parallel search* over multiple BFs (minimal overhead)
- What is considered “acceptable”, compare with network delay

	Local server	trans US	Europe
Ping time (ms)	0.227	90.615	110.978

Corpus size


Average Query Search Time for Different Database Sizes
0 Freq
Low Freq
Mid Freq
High Freq
Search time in ms
0
20
40
60
80
100
1000
5000
10000
20000
30000
50000
Number of documents in the database

OR improvement



Conclusion

- New search problem
- Efficient solution
- Introduction of a new encryption method
- Re-routable encryption primitive

Thank You!

- *Questions?*