Bolt: Data management for connected homes

Trinabh Gupta[*], Rayman Preet Singh[†]

Amar Phanishayee, Jaeyeon Jung, Ratul Mahajan

[*]The University of Texas at Austin

[†]University of Waterloo

Microsoft Research

Number of sensors, smart devices is growing

In 2008, number of sensors exceeded people

In 2020, 50 billion sensors.

http://share.cisco.com/internet-of-things.html

devices

In 2017, 90 million homes with automation

https://www.abiresearch.com/press/90-million-homes-worldwide-will-employ-home-automa

Automotive sensors

Sensors and devices for home automation

Need a **new** data management system for connected homes

PreHeat

[Ubicomp 2011]

DigiSwitch

[Medical Systems 2011]

Energy Data Analytics

[Energy and Building 2012]

Neighborhood Watch

[CSCW 2013]

Apps

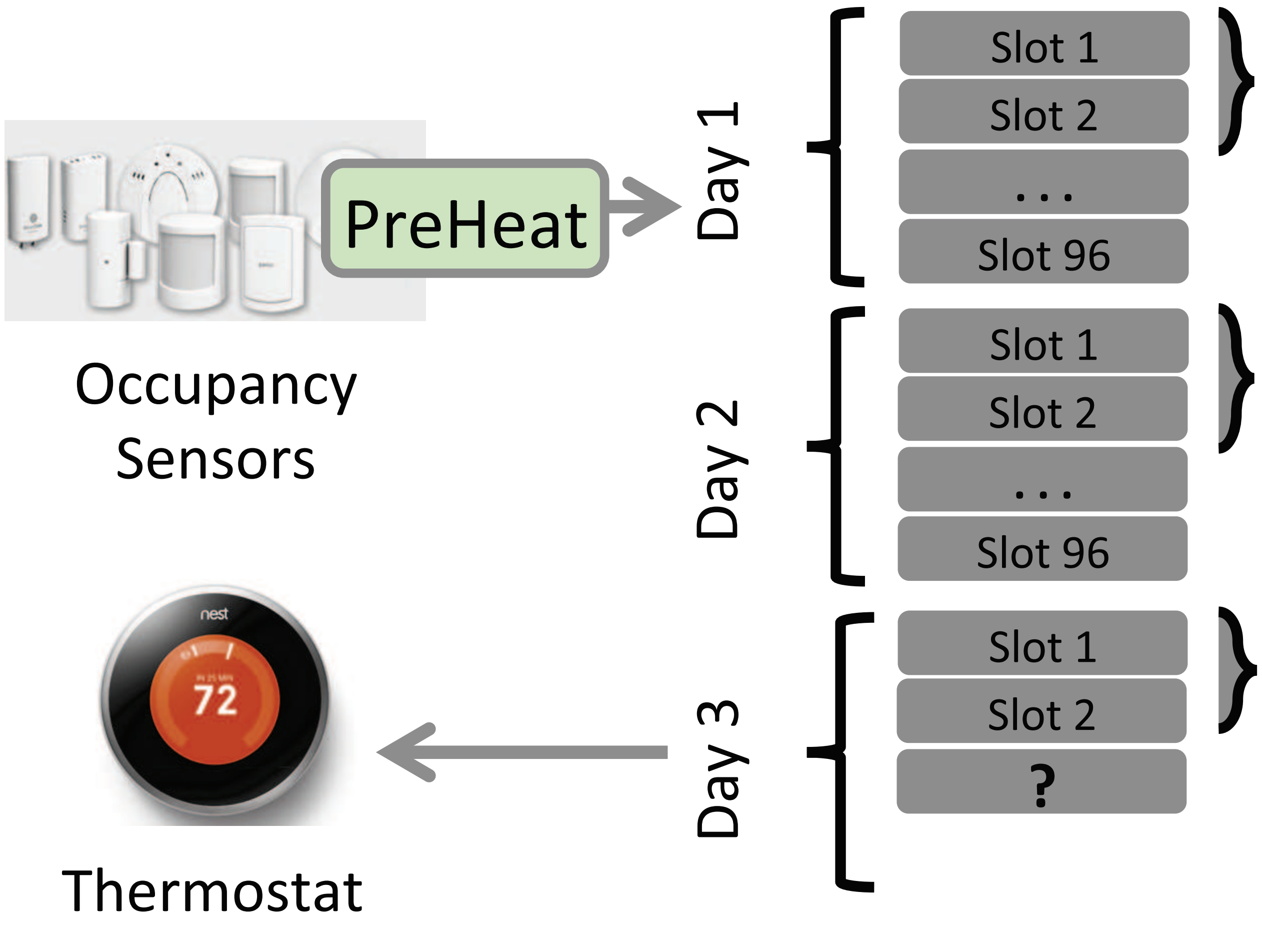
HomeOS

Mi Casa Verde

Platforms

Devices and sensors for the home

Applications generate time-series data and retrieve based on time windows



Identify days with closest occupancy pattern (*least hamming distance*) to predict future slot

Requirement: Support time-series data

Applications access data from multiple homes

Time	Attributes	Value
Mon, 1 AM	Temp = 20°C	0.9
Mon, 2 AM	Temp = 20°C	1.1
Mon, 3 AM	Temp = 22°C	1.2
Mon, 4 AM	Temp = 22°C	1.2

Energy Meter

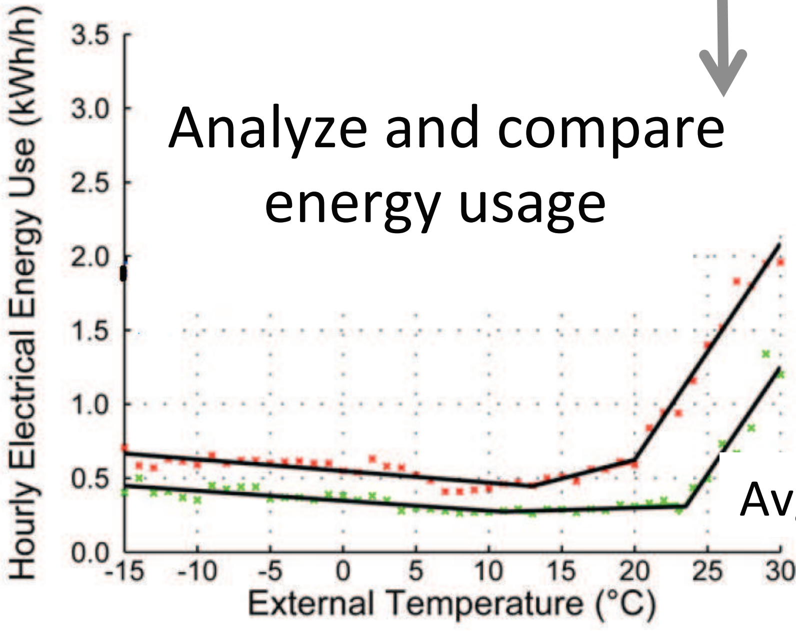
Energy Data Analytics

Run by utility company

Data from neighboring energy meters

Perform analysis even when homes are offline

Analyze and compare energy usage



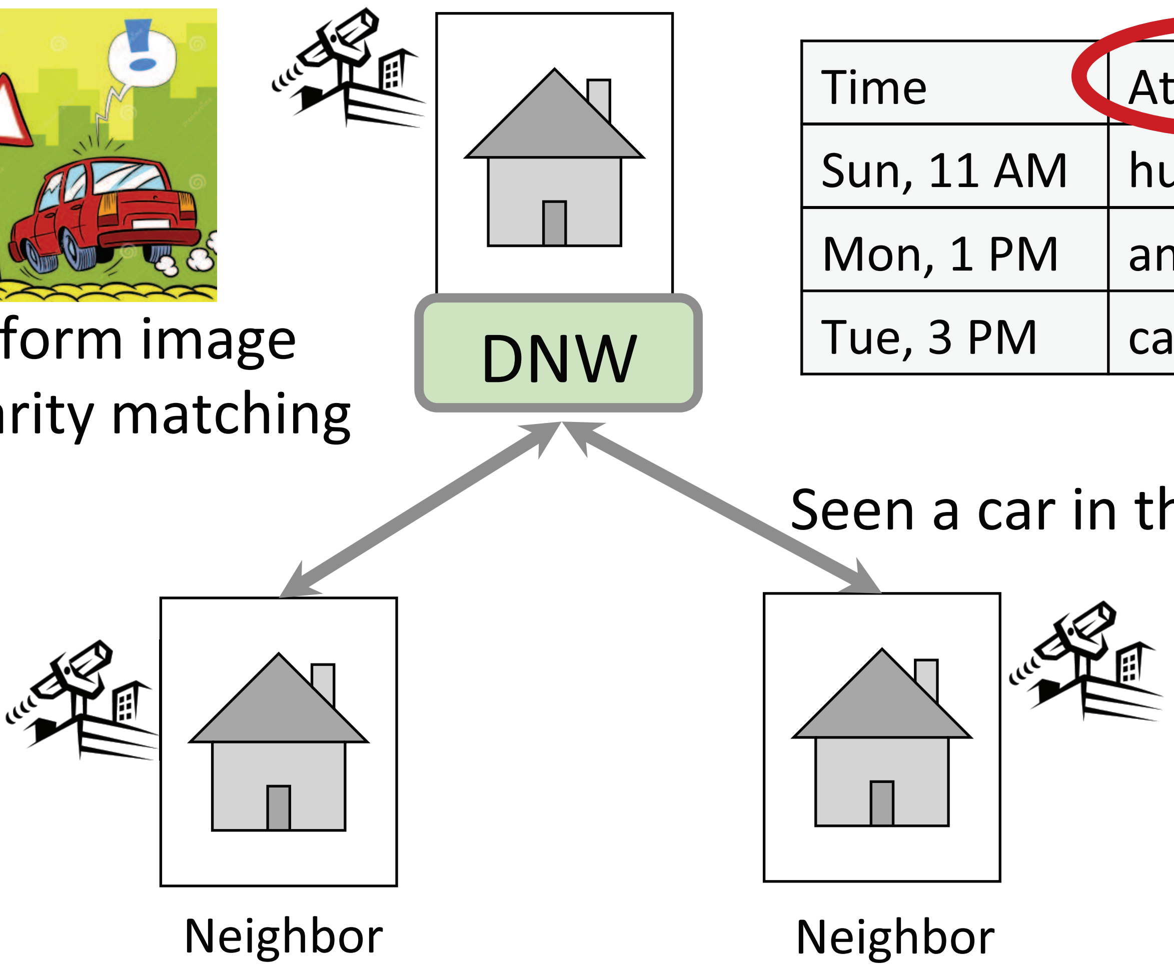
Avg. for this home

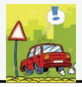
Avg. of neighboring homes

Requirement: Leverage cloud servers for availability

Applications share sensitive home data

Perform image similarity matching



Time	Attributes	Value
Sun, 11 AM	humans	
Mon, 1 PM	animal	
Tue, 3 PM	car, red	

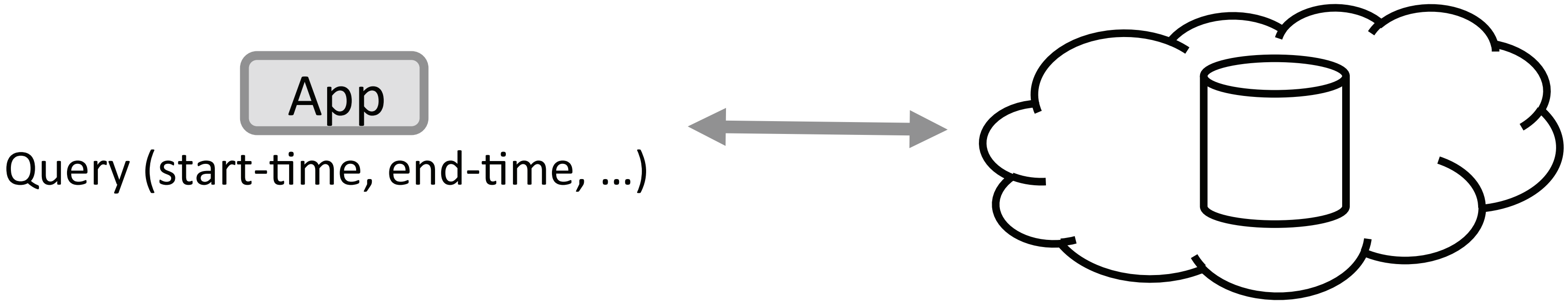
Seen a car in the last 24 hours?

Requirement: Ensure confidentiality, integrity

Recap of data management requirements

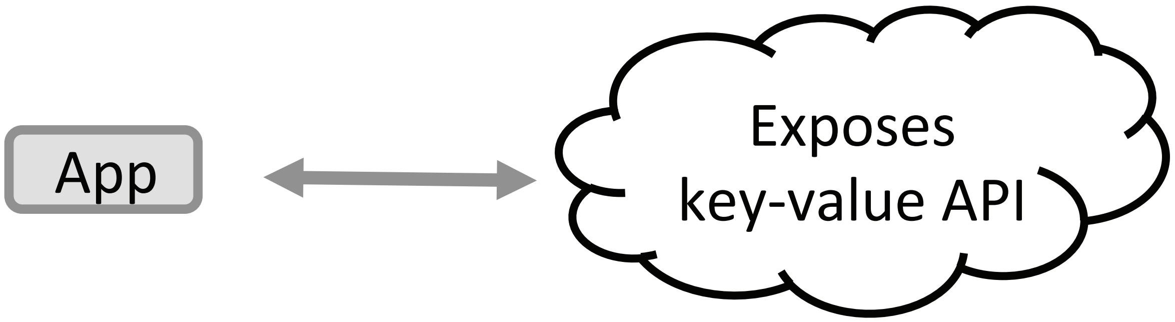
- Support time-series data with efficient time and tags based retrieval
- Leverage reliable and available cloud storage to facilitate sharing
- Ensure data confidentiality and integrity

Existing systems are not suitable



Time series data processing [OpenTSDB]

- *Do not maintain confidentiality or integrity of data*



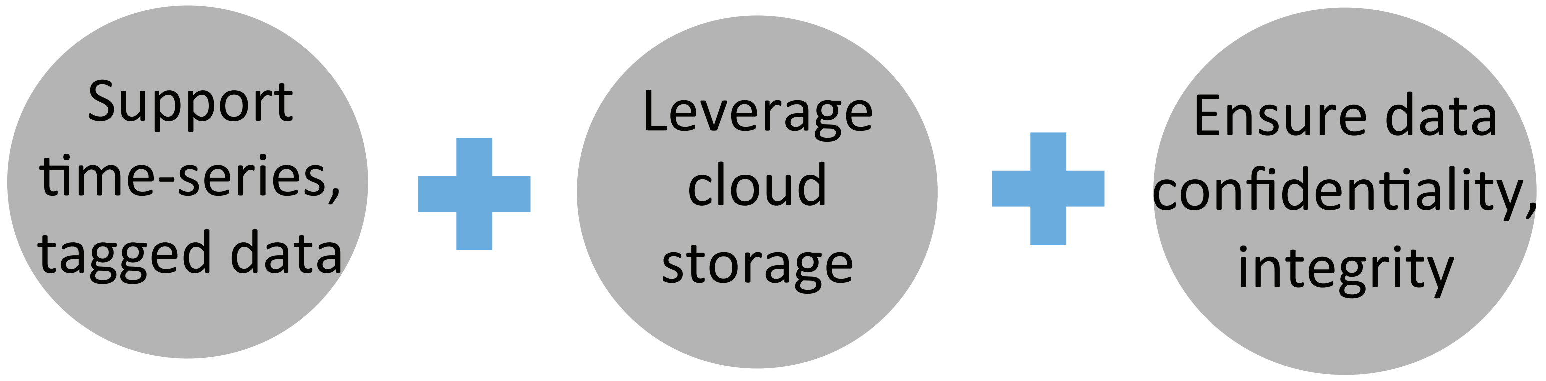
Secure systems using untrusted storage [SUNDR 04, Depot 10, SPORC 10]

- *Do not support time-series data*

Outline

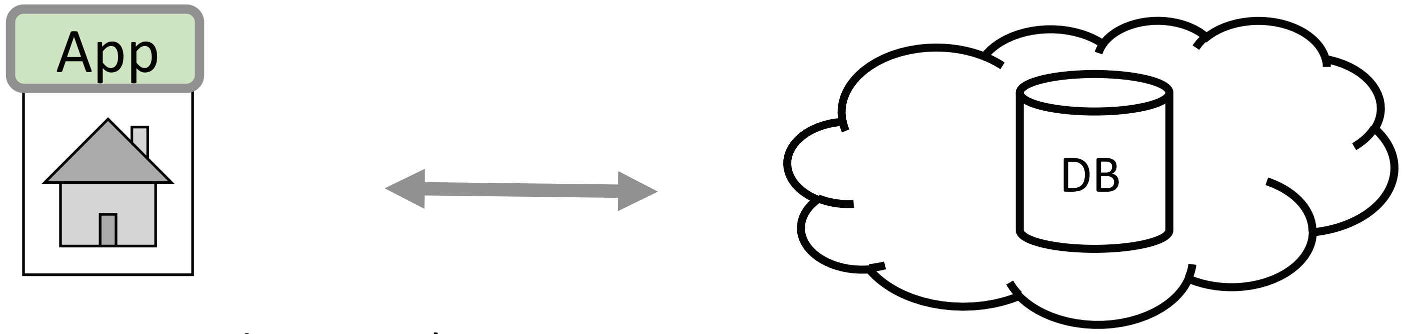
- Applications requirements and motivation
- Design of Bolt
  - Key mechanisms to support requirements
- Evaluation
  - Feasibility of using Bolt for three applications

Recall the data management requirements of apps for connected homes


Support time-series, tagged data
Leverage cloud storage
Ensure data confidentiality, integrity

How can we address these requirements simultaneously?

Straw man: Store data in a cloud DB



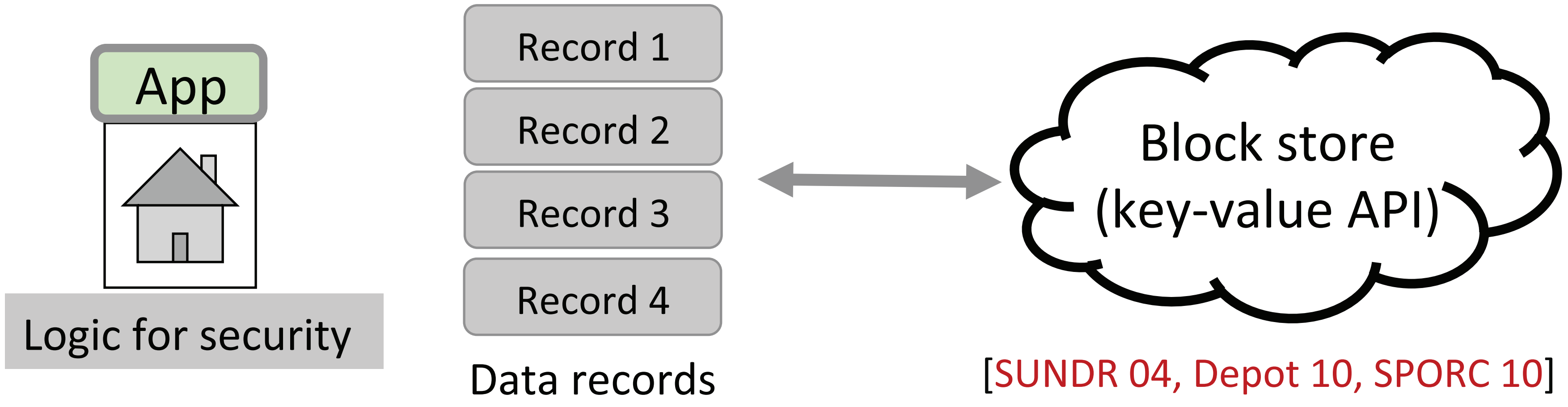
Query (start-time, end-time, …)

- Cloud untrusted for data confidentiality and integrity
- Cloud untrusted for computations (e.g., hamming distance, image similarity)

Design guidelines:

1. *End-points perform: encryption/decryption, data integrity checking, query evaluation*
2. *Use cloud providers for (just) storage*

Straw man: Using secure key-value datastores



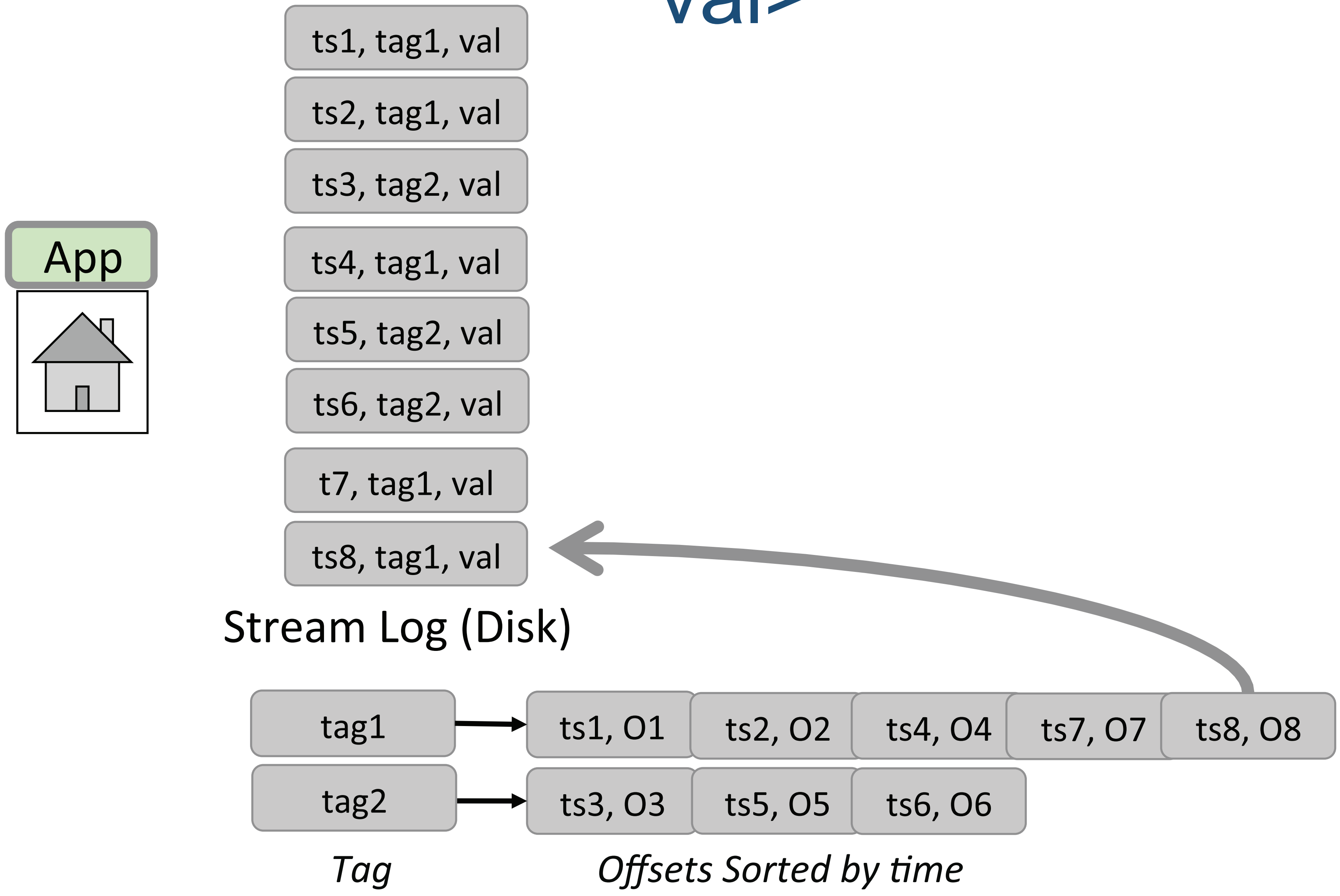
- Need support for temporal queries.
- High per-data-record overhead.
  - Encryption/decryption, integrity metadata / checks
  - Remote storage calls and transfers
- Individual data records do not compress well.

Design guideline: Batch contiguous data records, leverage workload query pattern

Overview of Bolt

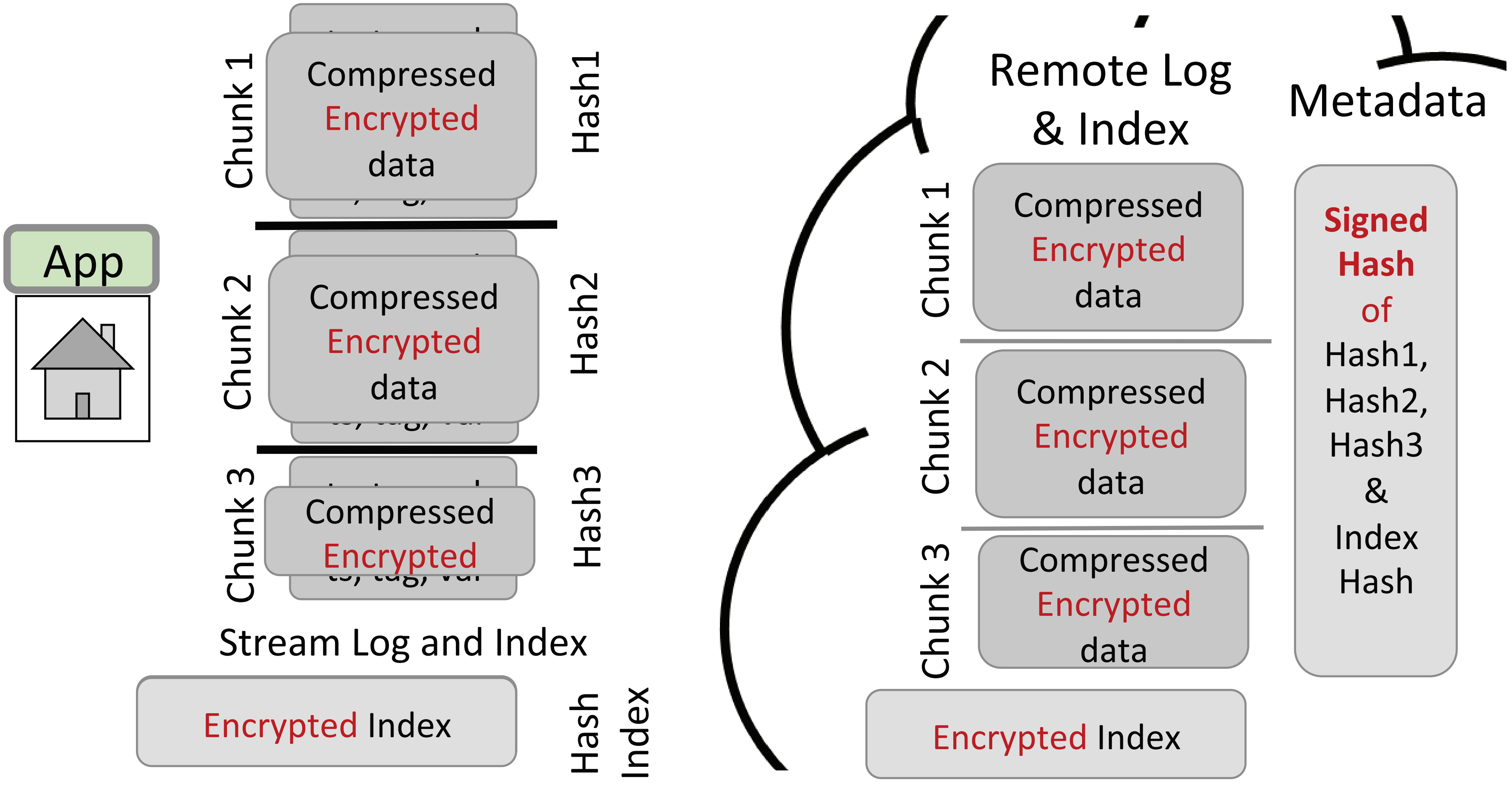
- Stream (append-only) abstraction
    - Records: <timestamp, [tag], value>
- Query (start-time, end-time, tag)
- Leverage cloud storage
    - Cloud resources untrusted for compute and storage
    - No cloud query engine with computation at endpoints
- Security and privacy guarantees
    - Confidentiality, Tamper evidence, Freshness

Bolt Stream: Index + Log of <ts, tag, val>



Stream Index (In memory, Disk backed)

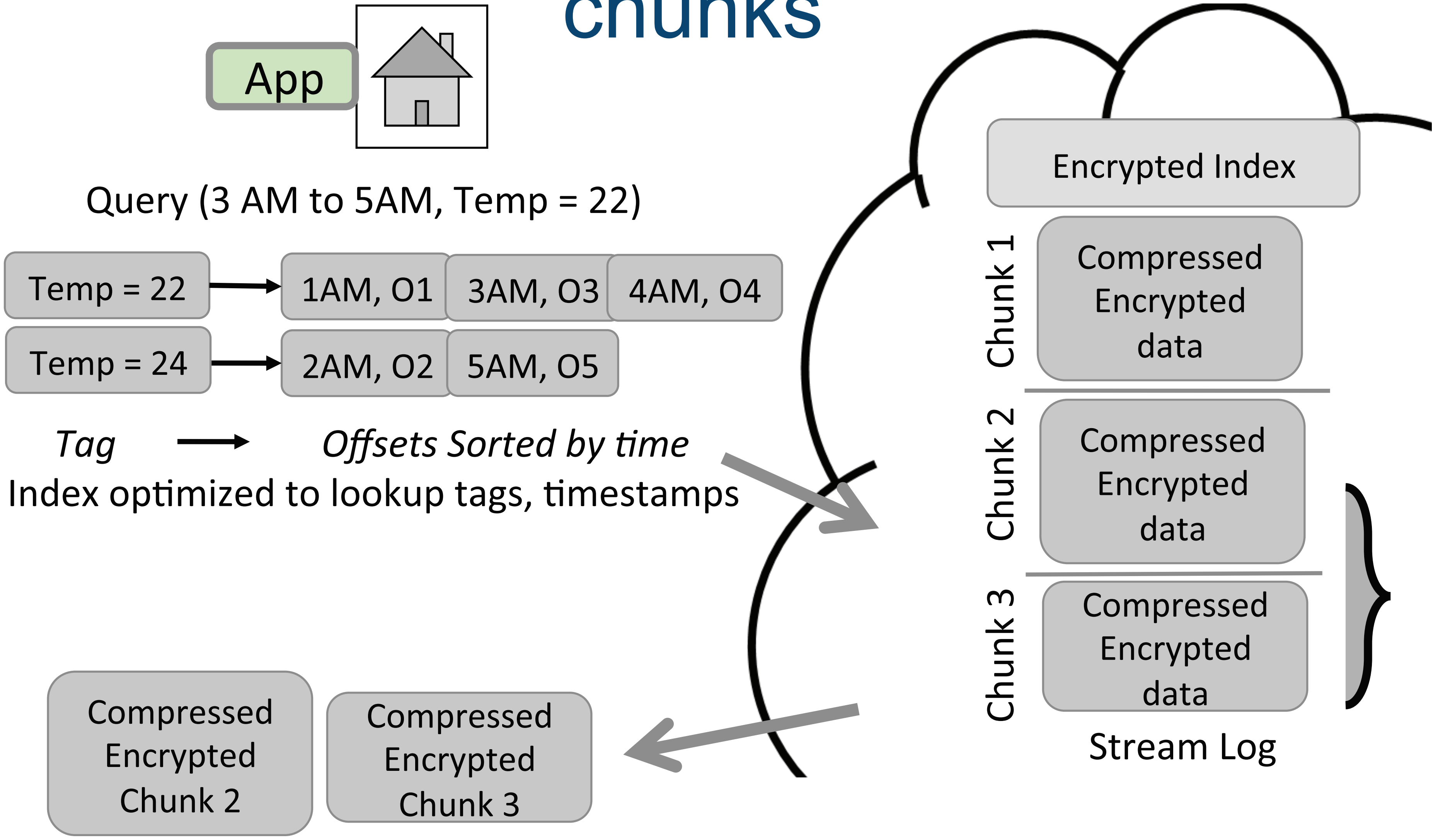
Batching data for efficiency



Improves storage and transfer efficiency.

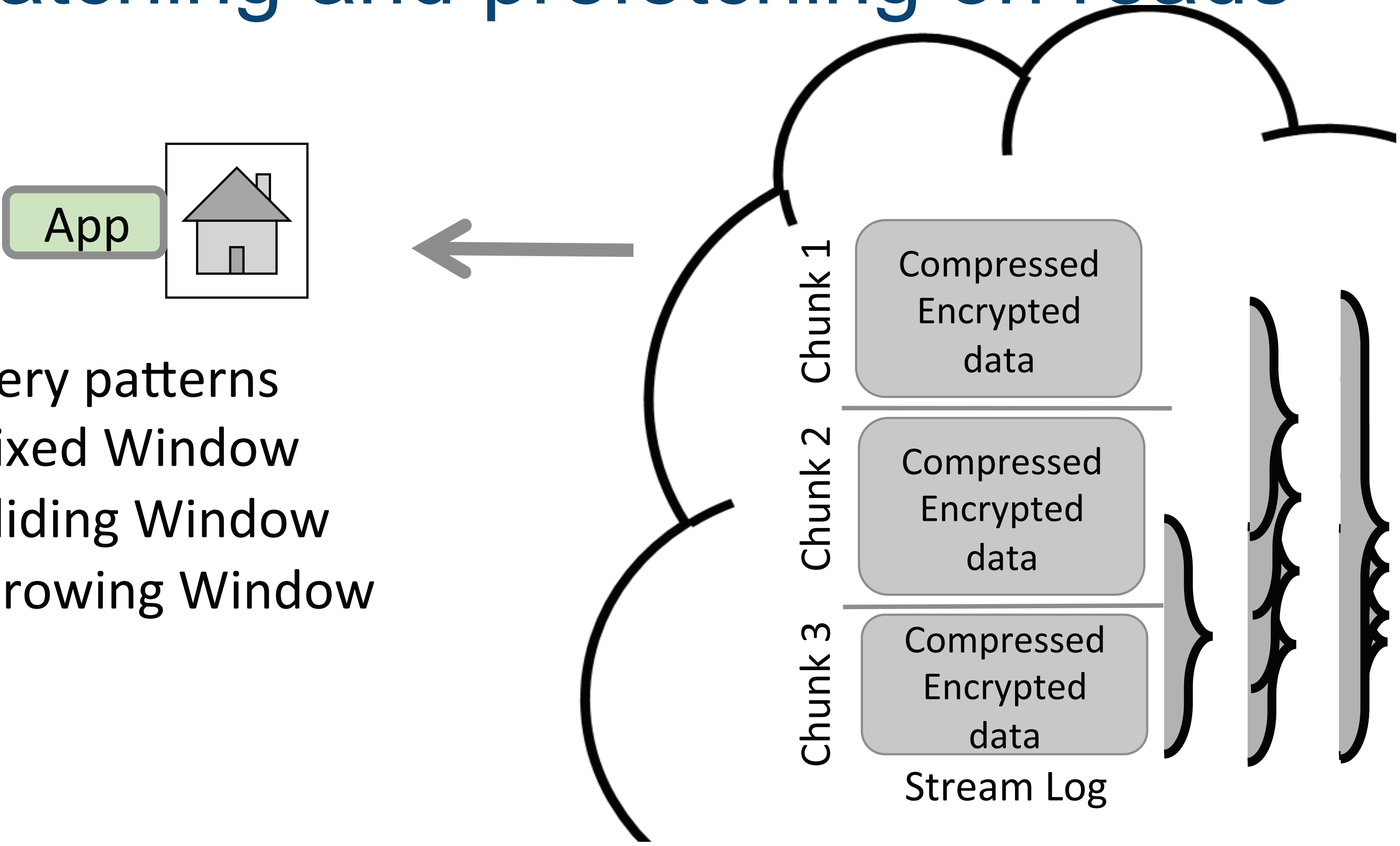
Amortizes cost of compression, encryption, and hashing

Reads use the index to download chunks


App
Query (3 AM to 5AM, Temp = 22)
Temp = 22
1AM, O1
3AM, O3
4AM, O4
Temp = 24
2AM, O2
5AM, O5
Tag
Offsets Sorted by time
Index optimized to lookup tags, timestamps
Encrypted Index
Chunk 1
Compressed Encrypted data
Chunk 2
Compressed Encrypted data
Chunk 3
Compressed Encrypted data
Stream Log
Compressed Encrypted Chunk 2
Compressed Encrypted Chunk 3

Lookups and computation are performed locally at home

Batching and prefetching on reads

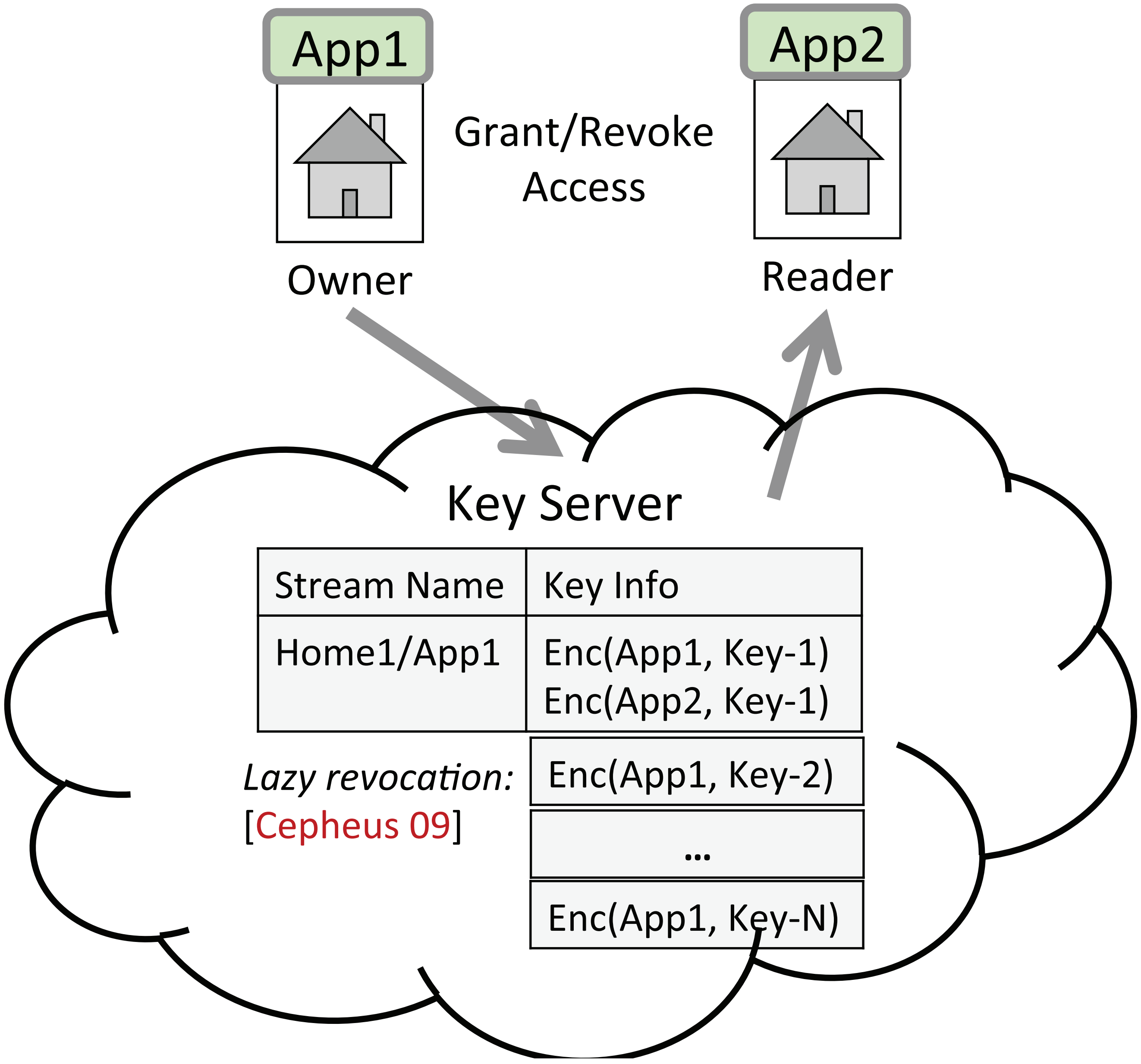

App
Chunk 1
Compressed Encrypted data
Chunk 2
Compressed Encrypted data
Chunk 3
Compressed Encrypted data
Stream Log

Query patterns

- Fixed Window
- Sliding Window
- Growing Window

Reduces number of remote calls, pre-fetches data for subsequent queries.

Secure sharing: Decentralized access control



Stream Name	Key Info
Home1/App1	Enc(App1, Key-1) Enc(App2, Key-1)

Enc(App1, Key-2)
…
Enc(App1, Key-N)

Addressing challenges in decentralized access control

- Potentially many encryption keys per stream.

  Solution: Hash-based key regression [Fu et al. NDSS 06]

- Key server trusted to maintain principal -> public key mappings.

- Key server trusted to prevent rollback of key.

  Possible solution: Replicated key server

Outline

- Applications requirements and motivation

- Design and key mechanisms of Bolt
  - Chunking
  - Separation of Index from data
  - Decentralized access control
  - Segmentation for memory efficiency, key change (paper)

- Evaluation
  - Feasibility of using Bolt for real-world applications

Implementation

- Integrated with HomeOS
  - labofthings.codeplex.com

- Supports Windows Azure and Amazon S3

- Integrated Bolt with 5 applications
  - 2 of these done by other developers
  - In use by HCI Researchers at MSR and Univ. of Michigan

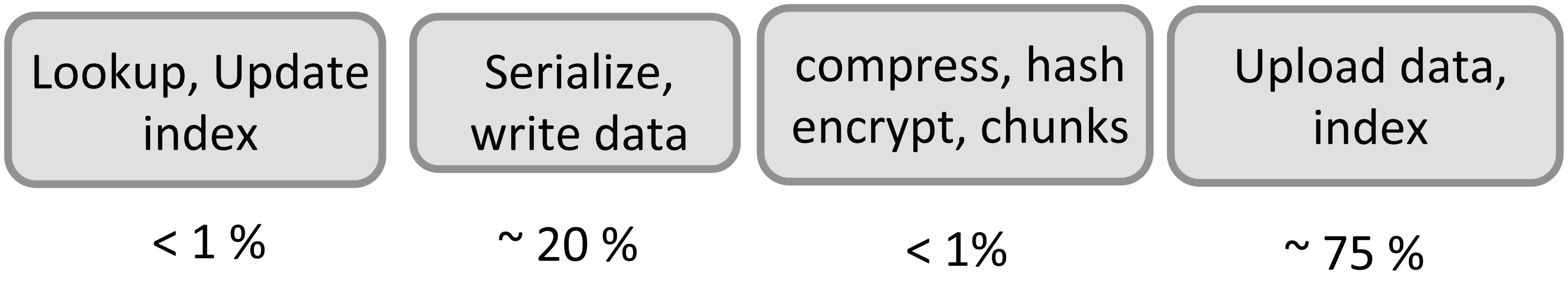
What are the overheads in Bolt?

- Baseline: Flat file
  - No support for temporal range queries, security

- Experiment to understand
  - Query time breakup
  - Storage overhead

Overheads in Bolt

Append (Temp = 22, Val = 0.7)



- Lookup during queries has < 1% overhead.
- Encryption, hashing overhead is negligible.

- Index storage adds
  - 30% for datavalue sizes of 10 bytes
  - < 1% for datavalue sizes of 1KB

Refer to paper for detailed microbenchmarks

Energy Data Analytics

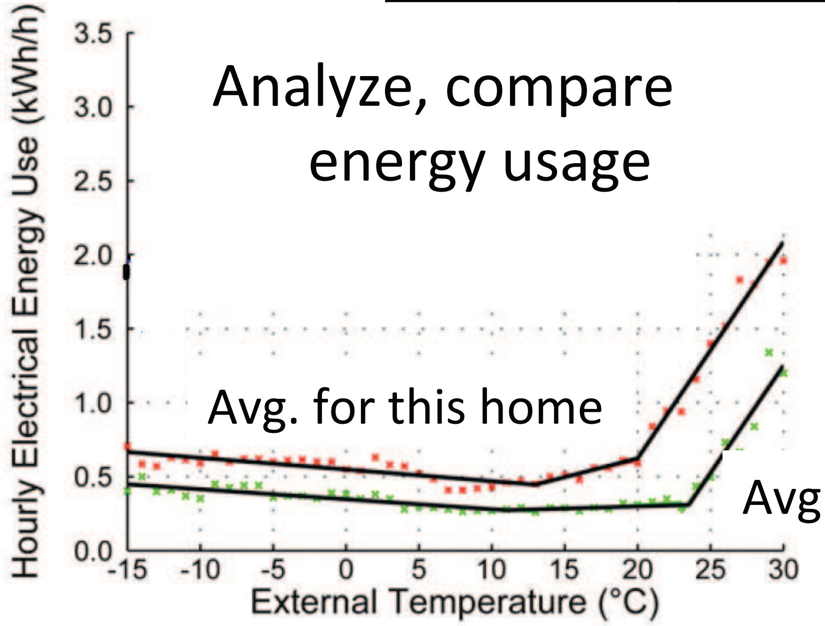
Energy Data Analytics

Energy Meter

Time	Attributes	Value
Mon, 1 AM	Temp = 20°C	0.9
Mon, 2 AM	Temp = 20°C	1.1
…	…	…
Tue, 4 AM	Temp = 22°C	1.2

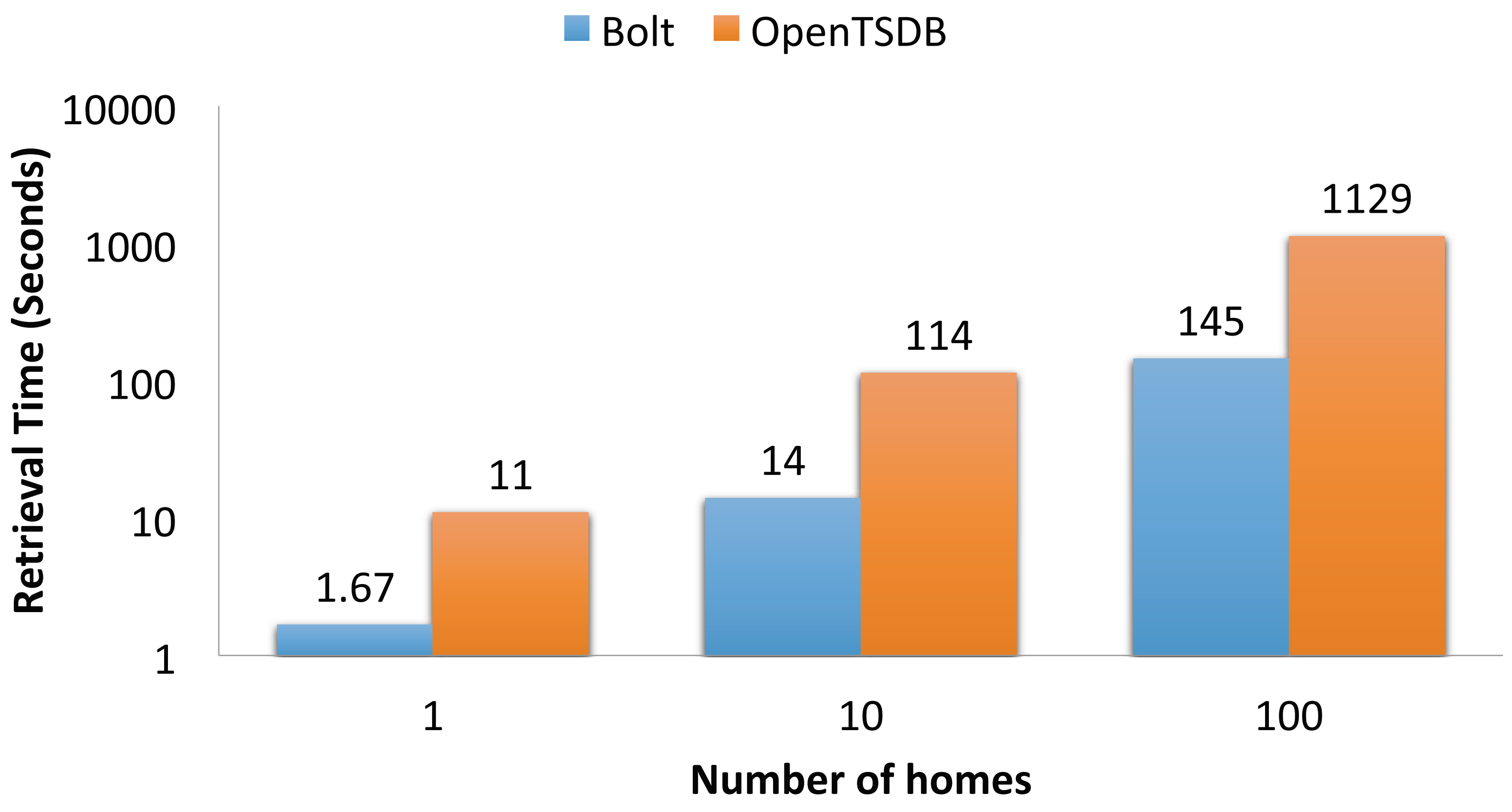
Energy reading for last 30 days when external temperature was X°C

Analyze, compare energy usage



Measure time taken to compare energy usage during last 30 days

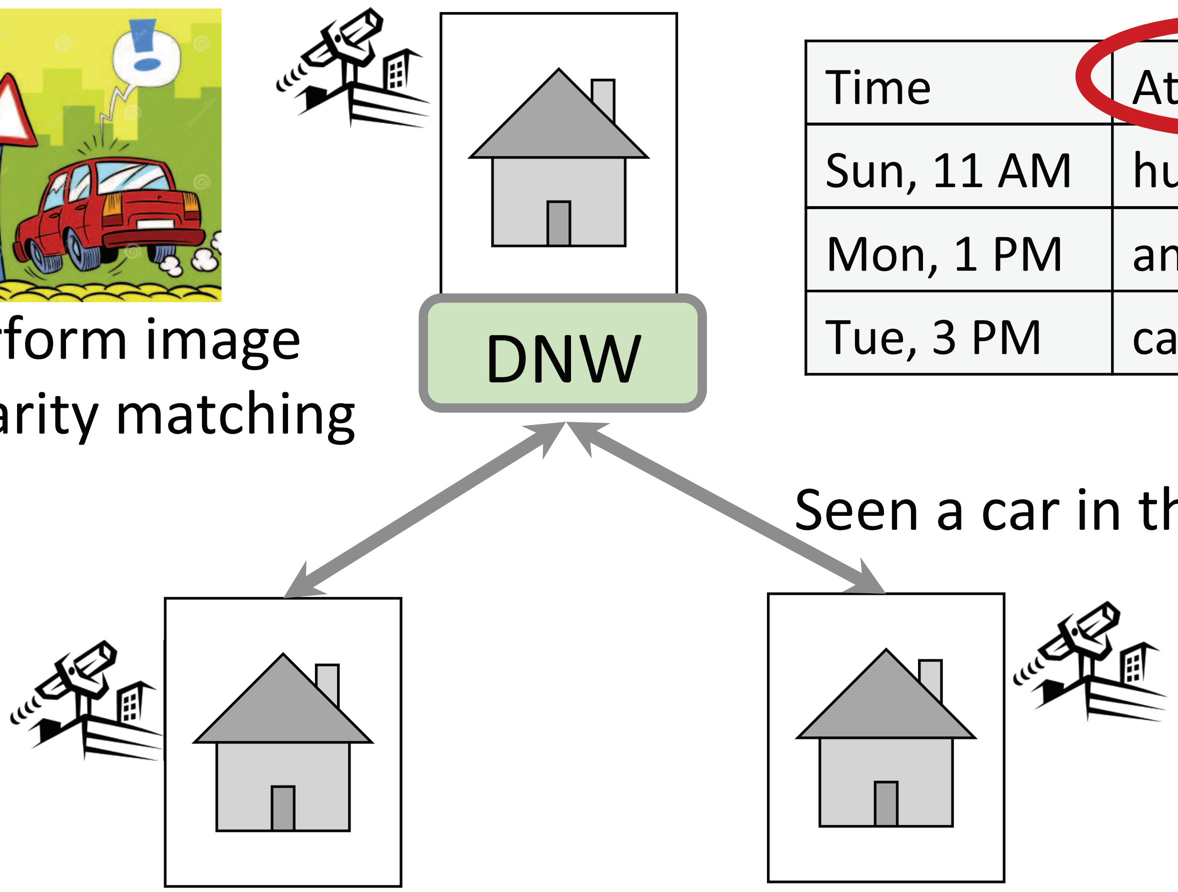
Prefetching in chunks improves query latency

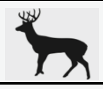
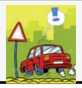


Current query retrieves data for subsequent query's temperature values

Applications share sensitive home data

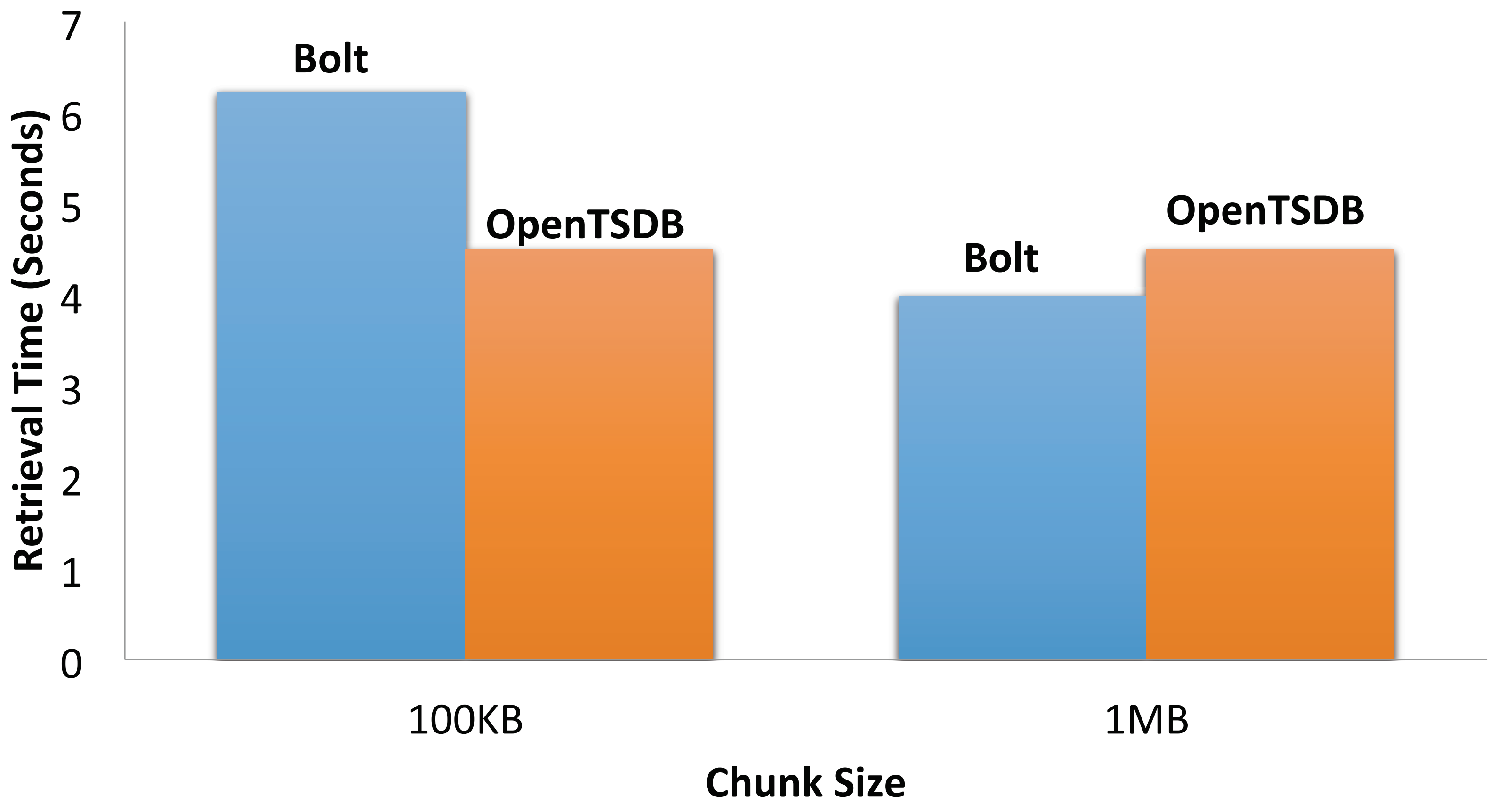
Perform image similarity matching



Time	Attributes	Value
Sun, 11 AM	humans	
Mon, 1 PM	animal	
Tue, 3 PM	car, red	

Measure query time across 10 homes looking at data from last 10 hours

Batching data in chunks improves query latency



Larger chunks result in fewer remote calls & RTTs.

Bolt’s data storage efficiency

	Bolt	OpenTSDB
Preheat	1.5	8.2
DNW	37.9	212.4
EDA	4.6	14.4

Data in MBs

Bolt is 3-5x more space efficient than OpenTSDB.

Summary

- Emerging class of applications for smart homes with a new set of data management requirements.

- Bolt addresses these efficiently by leveraging the nature of queries in this domain.

- Despite providing more than OpenTSDB (security guarantees), Bolt is up to 40x faster while requiring 3–5x less storage space.

Code: ***labofthings.codeplex.com***