Sieve: Cryptographically Enforced Access Control for User Data in Untrusted Clouds

Frank Wang (MIT CSAIL), James Mickens (Harvard), Nickolai Zeldovich (MIT CSAIL), Vinod Vaikuntanathan (MIT CSAIL)
Motivation

FitBit Cloud Server

Boston Marathon

NY Marathon

Insurance
Motivation

FitBit Cloud Server

Boston Marathon
NY Marathon
Insurance
Motivation

FitBit Cloud Server

Boston Marathon
NY Marathon
Insurance
Motivation

FitBit Cloud Server

- type=race
- type=running
- type=fitness

Boston Marathon
NY Marathon
Insurance
Problem: Curious storage provider or external attacker

FitBit Cloud Server

type=race

type=running

type=fitness

Boston Marathon

NY Marathon

Insurance
Problem: Curious storage provider or external attacker

FitBit Cloud Server

- type=race
- type=running
- type=fitness

NY Marathon
Boston Marathon
Insurance
Naïve Approach: Encrypt Data under 1 key
Naïve Approach: Encrypt Data under 1 key
Naïve Approach: Encrypt Data under 1 key

FitBit Cloud Server

- type=race
- type=running
- type=fitness

Boston Marathon
NY Marathon
Insurance
Naïve Approach: Encrypt Data under 1 key

How does the user selectively disclose her data?
Contributions

• **Sieve:** a new platform that allows users to *selectively* and *securely* disclose their data
  – Sieve protects against server compromise
  – Sieve hides key management from users
  – Reasonable performance
  – Sieve supports revocation
  – Good for web services that analyze user data
Outline

• Sieve
  – Protocol
  – Optimizations
  – Revocation
• Implementation
• Evaluation
Sieve Overview
Sieve Overview

User

Storage Provider

Web services
Sieve Overview

User

Storage Provider

Web services

Sieve user client

Sieve storage daemon

Sieve data import
Sieve Overview

User

Storage Provider

Web services

Sieve user client

Sieve storage daemon

Sieve data import

Location=US,
Year=2012,
Type=fitness

Year=2015,
Type=financial
Sieve Overview

User

Storage Provider

Web services

Sieve user client

Sieve storage daemon

Sieve data import

Location=US,
Year=2012,
Type=fitness

Year=2015,
Type=financial
Sieve Overview

User

Sieve user client

(Year < 2013 AND Type=Fitness )

Storage Provider

Sieve storage daemon

Location=US, Year=2012, Type=fitness
Year=2015, Type=financial

Web services

Sieve data import

Sieve Overview

7
Sieve Overview

User

Storage Provider

Web services

Sieve user client

Sieve storage daemon

Sieve data import

Location=US,
Year=2012,
Type=fitness

Year=2015,
Type=financial

(Year < 2013 AND Type=Fitness)
Sieve Overview

User

Storage Provider

Sieve storage daemon

Web services

Sieve data import

Location=US,
Year=2012,
Type=fitness

Year=2015,
Type=financial

(Year < 2013 AND Type=Fitness)
Sieve Overview

User

Sieve user client

Storage Provider

Sieve storage daemon

Location=US,
Year=2012,
Type=fitness

Location=US,
Year=2015,
Type=fitness

Web services

Sieve data import

Location=US,
Year=2012,
Type=fitness

(Year < 2013 AND
Type=Fitness )
Sieve Overview

User

Sieve user client

Storage Provider

Sieve storage daemon

Location=US,
Year=2012,
Type=fitness
Year=2015,
Type=financial

Web services

Sieve data import

Location=US,
Year=2012,
Type=fitness

(Year < 2013 AND
Type=Fitness )
Threat Model

• Storage provider is a passive adversary
  – Adversary can read all data
  – Follows protocol

• Web services trusted with user data they are given access to

• User and her devices trusted
Our approach: Attribute-based encryption (ABE)

- Assume that user-specific ABE public/private key pair
- Three main functions
Our approach: Attribute-based encryption (ABE)

- Assume that user-specific ABE public/private key pair
- Three main functions

- GenerateDecKey
- Encrypt
- Decrypt
Our approach: Attribute-based encryption (ABE)

- Assume that user-specific ABE public/private key pair
- Three main functions

Policy: (Year < 2013 AND type=Fitness)

- GenerateDecKey
- Encrypt
- Decrypt
Our approach: Attribute-based encryption (ABE)

- Assume that user-specific ABE public/private key pair
- Three main functions

Policy: (Year < 2013 AND type=Fitness)
Our approach: Attribute-based encryption (ABE)

- Assume that user-specific ABE public/private key pair
- Three main functions

Policy: (Year < 2013 AND type=Fitness)

Attributes: Location=US, Year=2012, Type=fitness

GenerateDecKey

Encrypt

Decrypt

(Year < 2013 AND Type=Fitness)
Our approach: Attribute-based encryption (ABE)

- Assume that user-specific ABE public/private key pair
- Three main functions

Policy: (Year < 2013 AND type=Fitness)

Attributes: Location=US, Year=2012, Type=fitness

GenerateDecKey

Encrypt

Decrypt
Our approach: Attribute-based encryption (ABE)

- Assume that user-specific ABE public/private key pair
- Three main functions
Our approach: Attribute-based encryption (ABE)

- Assume that user-specific ABE public/private key pair
- Three main functions

Policy: (Year < 2013 AND type=Fitness)

Attributes: Location=US, Year=2012, Type=fitness

GenerateDecKey

Encrypt

Decrypt

(Location=US, Year=2012, Type=fitness)

(Year < 2013 AND Type=Fitness)

(Location=US, Year=2012, Type=fitness)

(Year < 2013 AND type=Fitness)
Our approach: Attribute-based encryption (ABE)

- Assume that user-specific ABE public/private key pair
- Three main functions

Policy: $(\text{Year} < 2013 \text{ AND } \text{type}=\text{Fitness})$

Attributes: Location=US, Year=2012, Type=fitness

Note: attributes and policy are in cleartext
Sieve with ABE

User

Storage Provider

Web services

Sieve user client

Sieve storage daemon

Sieve data import
Sieve with ABE

User

Sieve user client

Location=US,
Year=2012,
Type=fitness

Year=2015,
Type=financial

Storage Provider

Sieve storage daemon

Web services

Sieve data import

ABE Encrypt
Sieve with ABE

User

Storage Provider

Web services

Sieve user client

Sieve storage daemon

Sieve data import

Location=US,
Year=2012,
Type=fitness

Year=2015,
Type=financial

ABE Encrypt
Sieve with ABE

User

Storage Provider

Web services

Sieve user client

Sieve storage daemon

Sieve data import

ABE Encrypt

(Year < 2013 AND Type=Fitness )

ABE GenerateDecKey

Location=US,
Year=2012,
Type=fitness

Year=2015,
Type=financial
Sieve with ABE

User

Storage Provider

Web services

Sieve user client

Sieve storage daemon

Sieve data import

ABE Encrypt

ABE GenerateDecKey

Location=US, Year=2012, Type=fitness

Year=2015, Type=financial

(Year < 2013 AND Type=Fitness )
Sieve with ABE

User

Storage Provider

Web services

Sieve user client

Sieve storage daemon

Sieve data import

ABE Encrypt

ABE GenerateDecKey

Location=US, Year=2012, Type=fitness

Year=2015, Type=financial

Location=US, Year=2012, Type=fitness

(Year < 2013 AND Type=Fitness)
Sieve with ABE

User

Storage Provider

Web services

Sieve user client

Sieve storage daemon

Sieve data import

ABE Encrypt

ABE GenerateDecKey

Location=US,
Year=2012,
Type=fitness

Location=US,
Year=2012,
Type=fitness

(Year < 2013 AND
Type=Fitness )

Year=2015,
Type=fitness

Year=2015,
Type=financial

Location=US,
Year=2012,
Type=fitness

ABE Decrypt
Challenges with ABE

• Performance
• Revocation
Reduce ABE Operations

• ABE is a public-key cryptosystem so slower than symmetric key cryptography

• Optimizations
  – Hybrid Encryption
  – Storage-based data structure
Reduce ABE Operations

• ABE is a public-key cryptosystem so slower than symmetric key cryptography

• Optimizations
  – Hybrid Encryption
  – Storage-based data structure
Hybrid Encryption
Hybrid Encryption
Hybrid Encryption
Hybrid Encryption

GUID

Data

symmetric

ABE

symmetric
Hybrid Encryption

Metadata block

GUID

symmetric

Data

ABE

symmetric
Hybrid Encryption

Metadata block

GUID

Data

symmetric

symmetric

Index
Attr1
Attr2
Attr3
Attr4
Attr5

Index
GUID1
GUID2
GUID3
GUID4
GUID5

meta
meta
meta
meta
meta

data
data
data
data

data

ABE
Hybrid Encryption

Metadata block

GUID

Data

symmetric

Index
Attr1 meta
Attr2 meta
Attr3 meta
Attr4 meta
Attr5 meta

Index
GUID1 data
GUID2 data
GUID3 data
GUID4 data
GUID5 data
Hybrid Encryption

Only have to perform symmetric key operations in future
Challenges with ABE

- Performance
- Revocation
Revocation

FitBit Cloud Server

- type=race
- type=running
- type=fitness

Boston Marathon
NY Marathon
Insurance

FitBit	Cloud	Server
Revocation

FitBit Cloud Server

- type=race
- type=running
- type=fitness

Boston Marathon
NY Marathon
Insurance
Revocation

FitBit Cloud Server

- type=race
- type=running
- type=fitness

Boston Marathon
NY Marathon
Insurance
Revocation

- Web service still has cached keys
- Need to re-encrypt data
Re-encryption with Hybrid Encryption

• Need to re-encrypt metadata and data
  – Easy to re-encrypt metadata block
  – How do we re-encrypt data object?
    • Download, re-encrypt, and upload
    • Requires substantial bandwidth and client-side computation
Solution: Key Homomorphism

• Allows changing key in encrypted data
  – Symmetric cipher that provides *in-place* re-encryption

• Does not learn old key, new key, or plaintext

• More specifics on scheme are in the paper
Full Revocation Process

Metadata Block

 symmetric

 ABE (attrs, epoch = 0)

Data

 symmetric

18
Full Revocation Process

Metadata Block

symmetric

Data

ABE (attrs, epoch = 0)
symmetric
Full Revocation Process

Metadata Block

Data

\[ \delta(\text{.attrs, epoch = 0}) \]
Full Revocation Process

Metadata Block

symmetric

Data

δ

ABE (attrs, epoch = 0)

symmetric
Full Revocation Process

Metadata Block

 symmetric

ABE (attrs, epoch = 0)

Data

 symmetric

 symmetric
Full Revocation Process

Metadata Block

Data

 symmetric

 symmetric

 symmetric

 symmetric

 symmetric

 symmetric

 symmetric

 symmetric

 ABE (attrs, epoch = 0)

 ABE (attrs, epoch = 1)
Full Revocation Process

Metadata Block

Data

ABE (attrs, epoch = 0)

ABE (attrs, epoch = 1)

symmetric

symmetric

symmetric
Issue new keys to web services whose data access has been changed and affected by revocation.
Outline

• Sieve
  – Protocol
  – Optimizations
  – Revocation

• Implementation

• Evaluation
Sieve Implementation

Cryptography:
• Libfenc with Stanford PBC for ABE
• AES (no revocation) and randomized counter mode with Ed448 (revocation)
Sieve Implementation

Cryptography:
- Libfenc with Stanford PBC for ABE
- AES (no revocation) and randomized counter mode with Ed448 (revocation)

User
  - Sieve user client
    - ~1400 LoC

Storage Provider
  - Sieve storage daemon
    - ~1000 LoC
    - MongoDB and BerkeleyDB

Web services
  - Sieve data import
    - Service-specific
Evaluation

• Is it easy to integrate Sieve into existing web services?
• Can web services achieve reasonable performance while using Sieve?
Evaluation Setup

- Multicore machine, 2.4 GHz Intel Xeon
- Web servers ran on machine’s loopback
  - Minimize network latency
  - Focus on cryptographic overheads
Case Studies

• Integrated with 2 open source web services
  – Open mHealth, health: small data
    • Visualize health data
    • One week’s health data: 6 KB
  – Piwigo, photo: large data
    • Edit and display photos
    • One photo: 375 KB
Easy to integrate with Sieve

• Lines of code required for integration
  – Open mHealth: ~ 200 lines
  – Piwigo: ~ 250 lines
Acceptable performance for Open mHealth and Piwigo

Ed448 with key caching

<table>
<thead>
<tr>
<th></th>
<th>Seconds</th>
</tr>
</thead>
<tbody>
<tr>
<td>Open mHealth</td>
<td>0.5</td>
</tr>
<tr>
<td>Piwigo</td>
<td>6.0</td>
</tr>
</tbody>
</table>

- **Open mHealth**: 0.5 seconds
- **Piwigo**: 6.0 seconds

- **Write**
- **Read**
Performance gap between AES and Ed448

![Bar chart showing the performance gap between AES and Ed448 for Open mHealth](chart.png)
Server per-core throughput is good

• Open mHealth
  – Storage write: 50 MB/s
  – Web service import: 70 users/min (Ed448)

• Piwigo
  – Storage write: 200 MB/s
  – Web service import: 14 photos/min (Ed448)
Revocation performance is reasonable

• Re-encrypt a metadata block (10 attrs): 0.63 s
• Re-key 100 KB data block: 0.66 s
• Generate new 10 attribute key: 0.46 s
Summary

• Required < 250 LoC to integrate with case studies
• Read and write data in reasonable amount of time
• Good per-core server throughput for storage writes and web service data imports
• Revocation functions take < 1 second
Related Work

• Untrusted Servers
  – ShadowCrypt, SUNDR, Depot, SPORC, CryptDB, DepSky, Bstore, Mylar, Privly

• ABE and Predicate Encryption Storage
  – Persona, Priv.io, Catchet (ABE)
  – GORAM (Predicate)

• Access Delegation Schemes
  – OAuth, AAuth, Macaroons
Related Work

• Untrusted Servers
  Solve different problems than Sieve

• ABE and Predicate Encryption Storage
  – Persona, Priv.io, Catchet (ABE)
  – GORAM (Predicate)

• Access Delegation Schemes
  – OAuth, AAuth, Macaroons
Related Work

• Untrusted Servers
  - Solve different problems than Sieve

• ABE and Predicate Encryption Storage
  - No complete revocation and/or ability to recover from device loss

• Access Delegation Schemes
  - OAuth, AAuth, Macaroons
Related Work

- **Untrusted Servers**
  - Solve different problems than Sieve

- **ABE and Predicate Encryption Storage**
  - No complete revocation and/or ability to recover from device loss

- **Access Delegation Schemes**
  - Less secure and expressive than Sieve
Conclusions

• Sieve is a new access control system that allows users to selectively and securely expose their private cloud data to web services
• Efficiently use ABE to manage keys and policies
• Complete revocation scheme compatible with hybrid encryption using key homomorphism
• Easy to integrate and reasonable performance