Data protection by means of fragmentation
FOSAD 2016

Katarzyna KAPUSTA
Telecom ParisTech
katarzyna.kapusta@telecom-paristech.fr

September 1, 2016

Self introduction

- **PhD Student at Telecom ParisTech Universite Paris-Saclay**
  Supervisor: Gerard MEMMI, Funded by the ITEA2 CAP project
- Education :
  - M.Eng. Telecom ParisTech Universite Paris-Saclay, Paris, France
  - M.Sc. AGH University of Science and Technology, Cracow, Poland
- Previous work experience :
  - Security consultant, E&Y, Paris
  - Software developer intern at Thales Communications & Security, Paris
  - Software developer intern at CERN, Geneva

Outline of the presentation

1. Introduction: Why do we need fragmentation?
2. State of the art
    - Data fragmentation techniques
    - Academic and commercial systems using data fragmentation
3. Proposed keyless efficient algorithm for data fragmentation
    - Algorithm description
    - Security analysis
    - Performance results
4. Ongoing and future works

Introduction: Why do we need fragmentation?

- The security of encrypted data depends on the chosen algorithm, as well as on the strength and the secure storage of its key
- Fragmenting data into multiple fragments and dispersing these fragments over various locations aims at frustrating an attacker
- Nowadays, fragmentation is enabled by the cloud environment (large number of servers, multiple data centers)

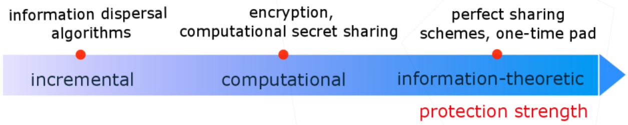
Part 1: State of the art

Our division of data fragmentation techniques

1. **Bitwise:** fragmenting data without any consideration for their structure, their semantics, or their uneven level of confidentiality
2. **Structurewise:** exploiting data structures, multi-level confidentiality, and machine trustworthiness

Bitwise fragmentation techniques and systems

- Three levels of security:
  - Perfect or information-theoretic security: (i.e. Shamir's secret sharing)
  - Computational security: standard encryption (i.e. AES)
  - Incremental security: Information Dispersal Algorithms (i.e. Rabin's)
- Challenge: balancing memory and performance with security
- Systems using bitwise fragmentation:
  - Academic, i.e. PASIS, POTSHARDS, GridSharing, DepSky
  - Commercial, i.e. Cleversafe (IBM), SecureParser (Unisys), Symform



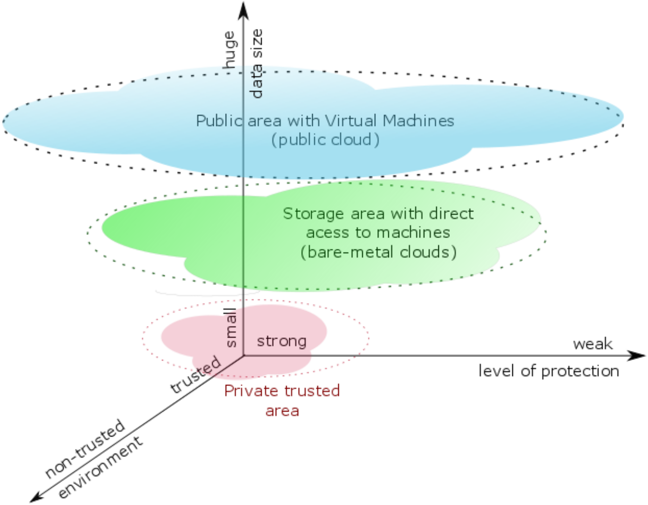
Structurewise fragmentation techniques

- Object-oriented
  - Fragmentation-Redundancy-Scattering
  - Breaking data into non-confidential fragments
  - Sensitive information encrypted and stored on trusted workstations, remaining pieces distributed over untrusted sites
- Database-oriented
  - Protecting relationships between relations
  - Preserving data unlinkability while executing queries
  - Searchable or partial encryption

Fragmentation in the cloud: issues and recommendations

1. Location control vs. virtualization
   - How to ensure secure data separation? Bare-metal clouds? Coarse-grained solution: multi-cloud
2. Latency problems: combining fragmentation with parallelization
3. Defining security levels without user interaction for fragmentation of structured data

Fragmentation in the cloud: desired architectural traits



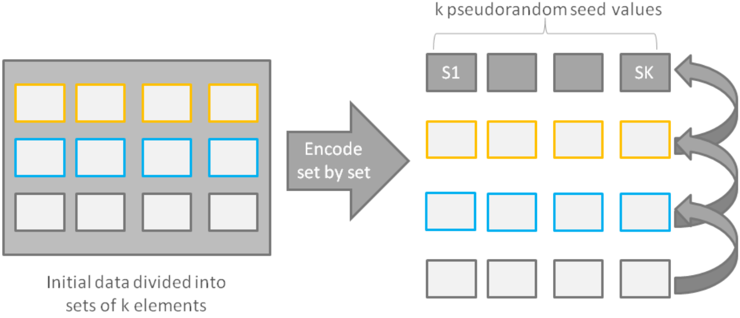
Part 2: Proposed fragmentation algorithm

Brief description of the idea

- **Problem** Perfectly secure fragmentation schemes increase memory, information dispersal algorithms have low security
- **Goal**: a fragmentation scheme balancing memory use and performance with security
- Proposal of a keyless computationally secure (k,n)-threshold algorithm:
  - 1st step: (k,k)-threshold fragmentation for security
  - 2st step: adding redundant fragments to obtain a (k,n)-threshold scheme

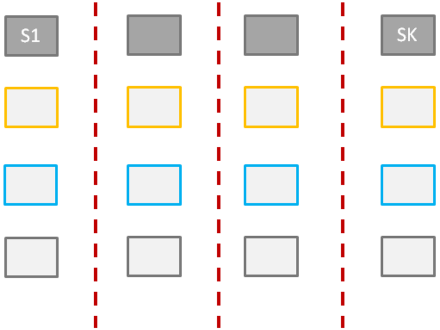
Principle of the fragmentation scheme (1)

- Initial data is divided into sets of k smaller data chunks
- Encoding done set by set in a Shamir like fashion
- Perfect security traded for memory: reusing encoding results
- A random seed of k values serves as the first set



Principle of the fragmentation scheme (2)

- Data fragmentation: encoded data are **separated** into k fragments
- **All or nothing**: all k fragments are needed for data recovery
- $n - k$ redundant fragments are added if needed



Characteristics

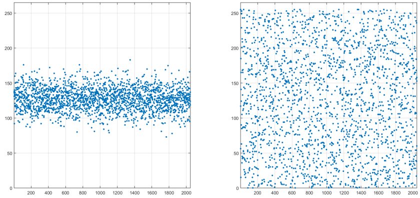
- **Memory use**: total overhead is of k bytes for one block of data, a fragment size is close to optimal value $\frac{D_{size}}{k}$
- **Performance**:
  - Fragmentation: $O(k)$ complexity, partially parallelizable
  - Defragmentation: complexity depends on the fragments used for recovery, highly parallelizable

Implementation

- **Matlab**: used for security analysis
- **JAVA**: single and 4-threaded version, multiple lookup tables, only logical operations (use of $GF(2^8)$), used for performance tests

Security analyses: fragments uniformity and independence

- Analyzing fragmentation results, comparing fragments to initial data
- **Uniformity**: chi-square test, data entropy, probability density function
- **Independence**: recurrence, correlation
- **Seed sensitivity**: same data fragmented using similar seeds



(a) (b)

Figure: Original data (a) and one of its fragment (b)

Security analyses: uniformity (1)

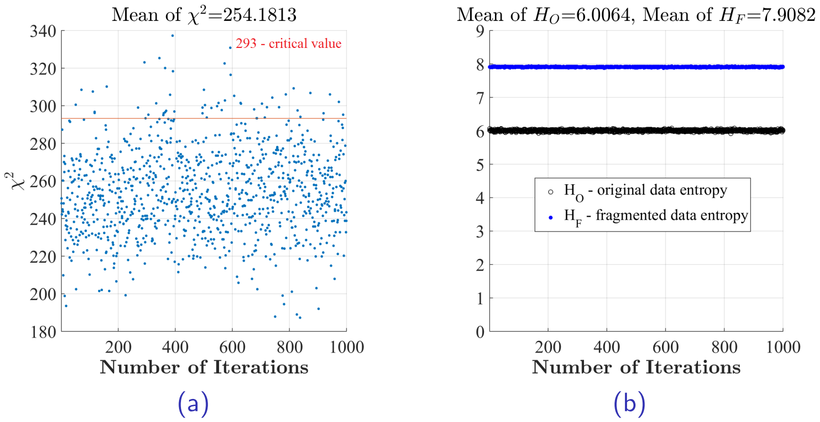
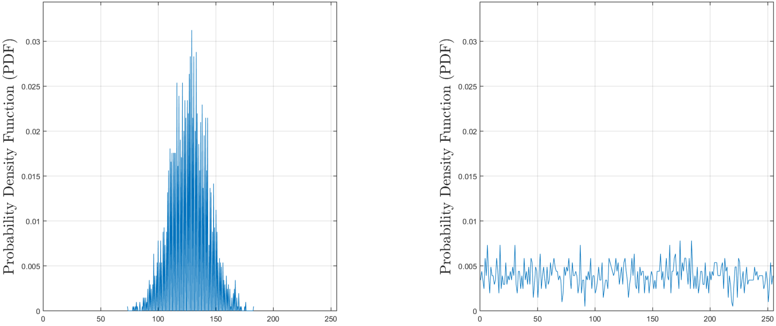


Figure: Chi-square test (a); Entropy comparison (b) ($k = 8$, for 1000 times)

Security analyses: uniformity (2)



(a) (b)

Figure: Probability Density Function of original data (a) and one of its fragment (b) ($k = 8$)

Security analyses: independence (1)

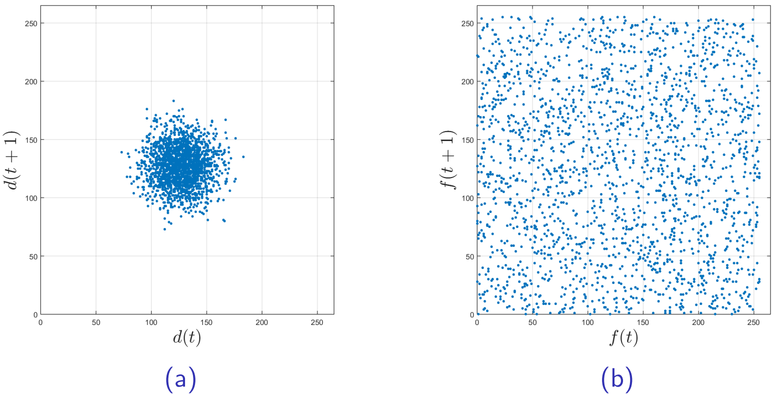
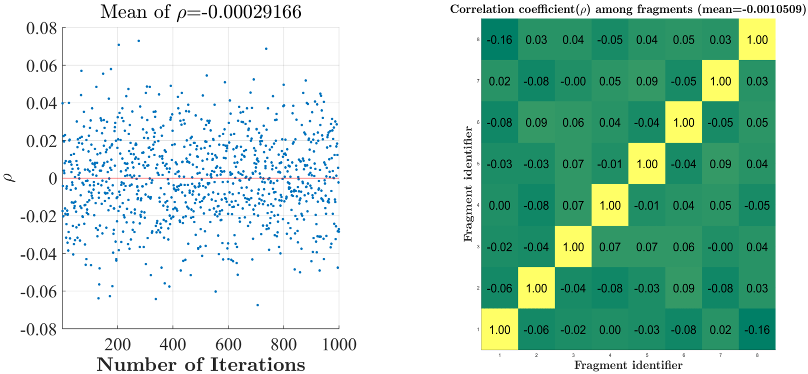


Figure: Recurrence plot of original data (a) and one of its fragment (b) ($k = 8$)

Security analyses: independence (2)

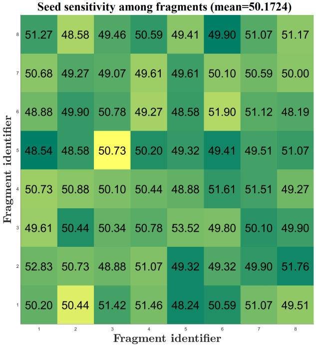


(a) (b)

Figure: Correlation coefficients between original data and its fragmentation ($k = 8$, for 1000 times) (a) and among fragments (b)

Security analyses: seed sensitivity

(a)



(b)

Figure: Correlations (a) and differences (b) between fragments of the same data fragmented with different seeds ($k = 8$)

Performance results

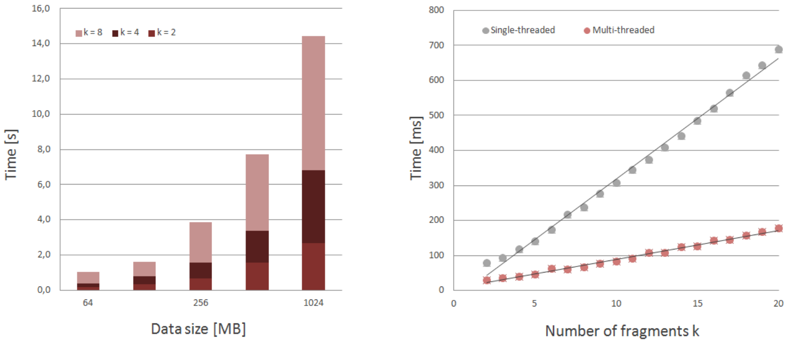


Figure: Time vs. number of fragments k (a), Time vs. data size (b).

Ongoing and future works

- Make our code open-source
- Benchmark the fragmentation scheme
- Refine the security analysis toolbox
- Adapt the fragmentation scheme to concrete use cases: cloud environment, unattended wireless sensor networks

Publications

- K. Kapusta, G. Memmi, and H.Noura, "POSTER: A Keyless Efficient Algorithm for Data Protection by Means of Fragmentation", in *ACM CCS 2016*, Vienna, 2016.
- K. Kapusta, P. Lambein, and G. Memmi, "POSTER: Data protection by means of fragmentation", in *RAID 2016*, Paris, 2016.
- K. Kapusta and G. Memmi, "Data protection by means of fragmentation in several distributed storage systems", in *CFIP-Notere*, Paris, 2015.
- G. Memmi, K.Kapusta, and H.Qiu, "Data protection by means of fragmentation in several distributed storage systems", in *Cyber Security of Smart Cities, Industrial Control System and Communications (SSIC)*, 2015
- G. Memmi, K.Kapusta, and H.Qiu, "Data Protection: Combining Fragmentation, Encryption, and Dispersion, an intermediary report", ITEA2-CAP WP3 Intermediary Report, June 2015.

Questions?