SparkFHE: Sharding for Batched Ciphertexts in Homomorphic Encryption

Student: Nabiha Raza  
nr6024@rit.edu

Advisor: Dr. Peizhao Hu  
ph@cs.rit.edu
Summary/Reminder of the Project

- Proposed/Accomplished Work
- Summary of Preliminary Results
  - Evaluation Metrics
  - Preliminary Results
  - Automating the Evaluation

- Conclusions
- Future Work
- References
Implement Sharding in Apache Spark.

Trying to achieve high fault tolerance if in case of disk failure.

End-to-End Encryption (FHE) with disk fault recovery have several applications.

- Patient Records
- Defense Data
- Voting Information
Formulated New Algorithm - “FHE-XCODE”

- X-Code with Fully Homomorphic Encryption
  - Combines XCode implemented so that it can be used in FHE
  - Additional Inexpensive operations on storage sets.

- Solves What?
  - Direct implementation is naive and computationally very expensive.
  - High reliability (storage recovery) with End-to-End Security (FHE).
  - Many applications where sensitive data is handled and manipulated (patient records, military records)
Summary of (Preliminary) Results

- Summary/Reminder of the Project
- Proposed/Accomplished Work
  - Summary of Preliminary Results
    - Evaluation Metrics
    - Preliminary Results
    - Automating the Evaluation
- Conclusions
- Future Work
- References
Evaluation Metrics

- Time taken to Fully Encode/Decode
- Time taken to execute individual steps/functions
- CPU utilization
- Memory utilization
- Test on multiple systems to check performance
- Access rate to the shards

Baseline Comparison with:
- Reed Solomon Codes
- STAR Codes
Evaluation Metrics

- Time taken to Fully Encode/Decode
- Time taken to execute individual steps/functions
- CPU utilization
- Memory utilization
- Test on multiple systems to check performance
- Access rate to the shards

Fig 1.0 Example: Execution Time taken by 7*7 Disk Size for complete recovery/decoding.
## Preliminary Results

<table>
<thead>
<tr>
<th>Disk Size</th>
<th>XCode (sec/min)</th>
<th>Reed Solomon (sec/min)</th>
</tr>
</thead>
<tbody>
<tr>
<td>7*7</td>
<td>6.53 seconds</td>
<td>10 seconds</td>
</tr>
<tr>
<td>13*13</td>
<td>9.74 seconds</td>
<td>11 seconds</td>
</tr>
<tr>
<td>97*97</td>
<td>109.2 sec (1.82 minutes)</td>
<td>130 seconds (2.16 minutes)</td>
</tr>
<tr>
<td>101*101</td>
<td>135 sec (2.25 minutes)</td>
<td>174 seconds (2.90 minutes)</td>
</tr>
<tr>
<td>479*479</td>
<td>671 sec (11.18 minutes)</td>
<td>800 seconds (13 minutes)</td>
</tr>
<tr>
<td>997*997</td>
<td>1417 sec (23 minute)</td>
<td>1621 seconds (27.01 minutes)</td>
</tr>
</tbody>
</table>

*Fig 2.0: Execution time taken for 2 Disk Recovery Comparison between XCode and RS Codes*
Preliminary Results

- Summary/Reminder of the Project
- Proposed/Accomplished Work
- Summary of Preliminary Results
- Preliminary Results
- Automating the Evaluation
- Conclusions
- Future Work
- References

**Fig 4.0** Comparison Trend for Execution Time taken 2 Disk Recovery Between XCode and Reed Solomon Codes

**Fig 5.0** Encoding Process Sample Tests taken and Averaged for 13*13 Matrix Disk Size
Automating the Evaluation

Python script to run systematically
- Will cover every possible value and store results in CSV for plotting (*Completed for Full Encoding and Decoding*)

**Fig 4.0** Python Script written to check for every combination till 2000*2000 Disks Sizes
Conclusions

Devised new Algorithm FHE-XCode

- Fault tolerant upto 2 Disks
- FHE-XCode provides:
  - Increased reliability
  - Increased security in the cloud.

Yes! It's been a success.
Further Extension

- Further Extension of **FHE-XCODE** for better memory utilization
- STAR Codes can be used along with FHE to recover three disks.
  - Increase the reliability.

Summary/Reminder of the Project
- Proposed/Accomplished Work
- Summary of Preliminary Results
  - Evaluation Metrics
  - Preliminary Results
  - Automating the Evaluation

Conclusions

References
References


Thank You

Any Questions?