Hyperledger Fabric Based Access Control
2019
Introduction

- **Name:** Sara Rouhani
- **Location:** Saskatoon, Canada
- **University:** University of Saskatchewan
- **Mentor(s):** Prof Rui Cruz, Rafael Belchior
- **Hyperledger project:** Hyperledger Fabric Based Access Control
Hyperledger Fabric Based Access Control

Project Description:
- Attribute based access control
- Hyperledger Fabric
- ArchiMate (Architecture Modeling)
- NodeJS API
- Hyperledger Caliper
- Matplotlib and Python
- HTML / CSS / Express / JavaScript
Hyperledger Fabric Based Access Control

Project Objectives:

- Obj 1: Designing the system architecture and application components
- Obj 2: Configuring Hyperledger Fabric
- Obj 3: Implementing the ChainCode
- Obj 4: Testing and analyzing system
- Obj 5: Writing the paper
- Obj 6: Throughout the documentation
Hyperledger Fabric Based Access Control

**Project Deliverables:**

- Deliverable 1: System architecture (document)
- Deliverable 2: ChainCode and application (code)
- Deliverable 3: Performance data based on Hyperledger Caliper (Data)
- Deliverable 4: Analyzing data and creating graphs
- Deliverable 4: Academic paper
- Deliverable 5: Documentation
Hyperledger Fabric Based Access Control

- **Project Execution & Accomplishments:**
  - ABAC application including ChainCode implementation and a simple web interface
  - Collecting performance analysis data based on Hyperledger caliper and different configuration
  - Data analysis and graphs design using Matplotlib library
  - Academic paper
  - Github Wiki page
Performance Graphs

- Kafka based on the number of transactions
Performance Graphs

Raft based on the number of transactions
Performance Graphs

- Kafka and Raft based on clients and average latency
Performance Graphs

- Kafka and raft based on send rate and average latency
Performance Graphs

Kafka and Raft based on send rate and throughput
Performance Graphs

The effect of increasing the number of peers and organizations and comparing two databases (CouchDB and goLevelDB)
## Resource consumption

<table>
<thead>
<tr>
<th>Orderer</th>
<th>Name</th>
<th>Memory(max)</th>
<th>Memory(avg)</th>
<th>CPU(max)</th>
<th>CPU(avg)</th>
<th>Traffic In</th>
<th>Traffic Out</th>
<th>Disc Read</th>
<th>Disc Write</th>
</tr>
</thead>
<tbody>
<tr>
<td>Raft</td>
<td>dev-peer0.org1</td>
<td>74.4MB</td>
<td>71.1MB</td>
<td>32.25%</td>
<td>28.21%</td>
<td>13.8MB</td>
<td>5.3MB</td>
<td>0B</td>
<td>0B</td>
</tr>
<tr>
<td></td>
<td>dev-peer0.org2</td>
<td>73.3MB</td>
<td>69.1MB</td>
<td>33.23%</td>
<td>28.28%</td>
<td>13.8MB</td>
<td>5.3MB</td>
<td>0B</td>
<td>0B</td>
</tr>
<tr>
<td></td>
<td>peer0.org1</td>
<td>379.3MB</td>
<td>369.4MB</td>
<td>67.21%</td>
<td>56.32%</td>
<td>31.1MB</td>
<td>23.7MB</td>
<td>0B</td>
<td>0B</td>
</tr>
<tr>
<td></td>
<td>peer0.org2</td>
<td>284.0MB</td>
<td>274.1MB</td>
<td>70.78%</td>
<td>55.66%</td>
<td>31.1MB</td>
<td>23.6MB</td>
<td>4.0KM</td>
<td>21.8MB</td>
</tr>
<tr>
<td></td>
<td>orderer1</td>
<td>554.1MB</td>
<td>535.4MB</td>
<td>26.11%</td>
<td>15.41%</td>
<td>22.4MB</td>
<td>59.1MB</td>
<td>0B</td>
<td>37.2MB</td>
</tr>
<tr>
<td></td>
<td>orderer2</td>
<td>525.3MB</td>
<td>506.5MB</td>
<td>18.78%</td>
<td>11.88%</td>
<td>27.6MB</td>
<td>28.7MB</td>
<td>0B</td>
<td>37.0MB</td>
</tr>
<tr>
<td></td>
<td>orderer0</td>
<td>513.3MB</td>
<td>494.7MB</td>
<td>19.40%</td>
<td>11.63%</td>
<td>27.5MB</td>
<td>10.6MB</td>
<td>0B</td>
<td>37.2MB</td>
</tr>
<tr>
<td>Kafka</td>
<td>dev-peer0.org1</td>
<td>73.5MB</td>
<td>72.8MB</td>
<td>17.87%</td>
<td>15.26%</td>
<td>7.5MB</td>
<td>2.5MB</td>
<td>0B</td>
<td>0B</td>
</tr>
<tr>
<td></td>
<td>dev-peer0.org2</td>
<td>64.5MB</td>
<td>62.8MB</td>
<td>18.73%</td>
<td>15.69%</td>
<td>7.5MB</td>
<td>2.5MB</td>
<td>0B</td>
<td>0B</td>
</tr>
<tr>
<td></td>
<td>peer0.org1</td>
<td>295.9MB</td>
<td>286.2MB</td>
<td>52.08%</td>
<td>49.15%</td>
<td>27.1MB</td>
<td>17.0MB</td>
<td>368.0KB</td>
<td>21.2MB</td>
</tr>
<tr>
<td></td>
<td>peer0.org2</td>
<td>294.1MB</td>
<td>282.7MB</td>
<td>51.54%</td>
<td>48.38%</td>
<td>27.1MB</td>
<td>17.1MB</td>
<td>152.0KB</td>
<td>21.2MB</td>
</tr>
<tr>
<td></td>
<td>orderer0</td>
<td>121.1MB</td>
<td>113.1MB</td>
<td>25.80%</td>
<td>23.30%</td>
<td>29.6MB</td>
<td>11.1MB</td>
<td>4.0KB</td>
<td>18.4MB</td>
</tr>
<tr>
<td></td>
<td>orderer1</td>
<td>124.1MB</td>
<td>115.2MB</td>
<td>21.87%</td>
<td>20.25%</td>
<td>29.7MB</td>
<td>46.5MB</td>
<td>276.0KB</td>
<td>18.4MB</td>
</tr>
</tbody>
</table>
Hyperledger Fabric Based Access Control

› Recommendations for future work:

❖ Present the project as a module for Hyperledger Fabric
❖ Integration with Hyperledger Indy