Microservice implementation of the fraud detection ring

Cheng Liang , email: a812789400@gmail.com [1]
S.V.Vlasov, email: svv@cs.vsu.ru [1]

[1] Voronezh State University

Аннотация. *In face of valuable loss of funds by financial organizations around the world, the easy accessible tool for detection of financial fraud and money laundering is been developed and provided as microservice and API for public use. In this paper test data were used to display architecture and interface of the microservice.*

Ключевые слова: *fraud detection, microservice, graph database, Neo4j, Docker, Kubernetes..*

Introduction

Digital processing of all financial operations and data become a normal in present days. Simultaneously the criminals activated a rather sophisticated methods of frauds and money laundering schemes. Group of investigators normally works in organization to deal with fraud detection [1]. The Artificial Intelligence and Machine Learning methods [2] greatly simplify and accelerate their work today.

Fraud detection system is based on the analysis of data and implements a data-driven design [3]. However, in this case the primary importance has been not the client data itself but rather connections between clients. Moreover, the argent response required to the detected fraud, that stimulates the event-driven [4] or behavior-driven [5] approach to the software development. The real time data flow provided by the financial organization can be served by remote service or API without intervention in secure and private data.

The architecture of the related software system developed by authors is based on the power of the graph database Neo4j [6] and implemented as a microservice with a few API in Docker [7]. To illustrate functionality of the system we are using artificial test data and orchestration facilities of Kubernetes [8].

1. Types of frauds and scenarios

There are many tricks used by fraudsters to achieve illegal income or privilege. It can be individual fraud with local bank account or insider information, but mostly it is an organized crime involving network of

individuals supporting the fraud. In particular money laundering defined in Wikipedia as following: “Money laundering is the process of changing large amounts of money obtained from crimes, such as drug trafficking, into origination from a legitimate source. It is a crime in many jurisdictions with varying definitions. It is a key operation of organized crime and the underground economy”. Thus, a detection of fraud rings becomes the most important part of the fraud and money laundering detection.

2. Neo4j graph database for detection of fraud ring

The graph database become active development field in recent decade. Graph database stores information about connections or relations between nodes. Different graph analytics methods provide possibility for detection special unordinary connections and clustering of nodes. There are number of popular graph database tools, such as Neo4j, ArangoDb, Dgraph, Cassandra, Apache Giraph to name a few. In the present research we are using Neo4j graph database for building a service for fraud detection ring.

3. Fraud detection service API using Docker and Kubernetes

The fraud detection to be useful in modern systems should be implemented as online service for streaming processing data in memory. The best approach is to use Docker containers as microservices for Neo4j data processing, graph analytics and fraud ring detection and reporting. Different.

4. Testing results and similar works

Production system proposed in this work currently under development and requires a deep testing. The importance of the automation and steaming of fraud detection attract great importance and discussed in multiple publications of different authors [8-10]. There is also another online implementation of fraud detection, for example, from Confluent [11] and Seon [12].

5. Fraud detection ring typical Scenario

While the details behind each fraudulent collusion vary by operation, the following patterns illustrate how fraudulent gangs generally operate:

Fraud gang of two or more people.

Ring members share a subset of legitimate contact information, i.e., phone numbers, addresses and passport numbers, combining them to create many fictitious identities.

Ring members use these fictitious identities to open accounts.

New accounts are added to the original accounts: unsecured lines of credit, credit cards, personal loans, etc.

Account in normal use, regular purchases, and timely payments.

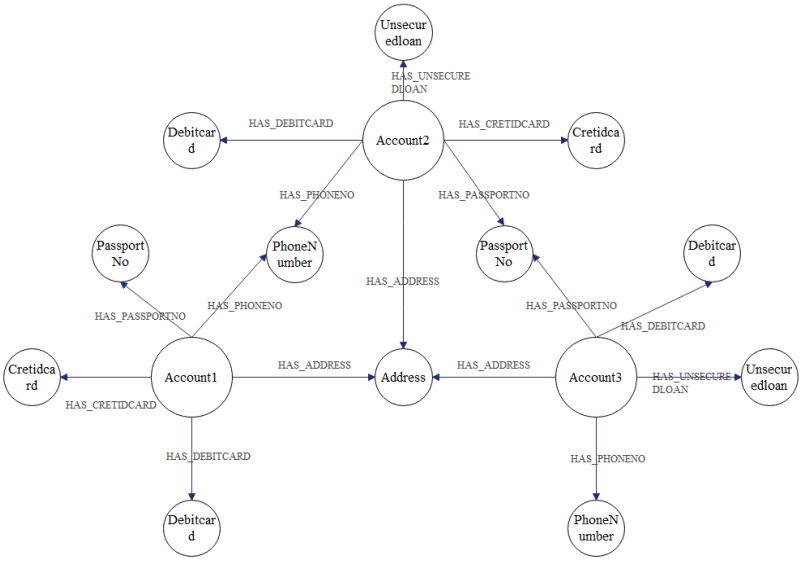
Banks increase credit lines over time as they observe normal credit behavior.

One day, swap members max out all their credits, then disappear.

Sometimes fraudsters go a step further and use fake checks to zero out all balances, doubling their losses.

The collection process ensues, but the agent can never reach the fraudster.

Uncollectible debts are written off.



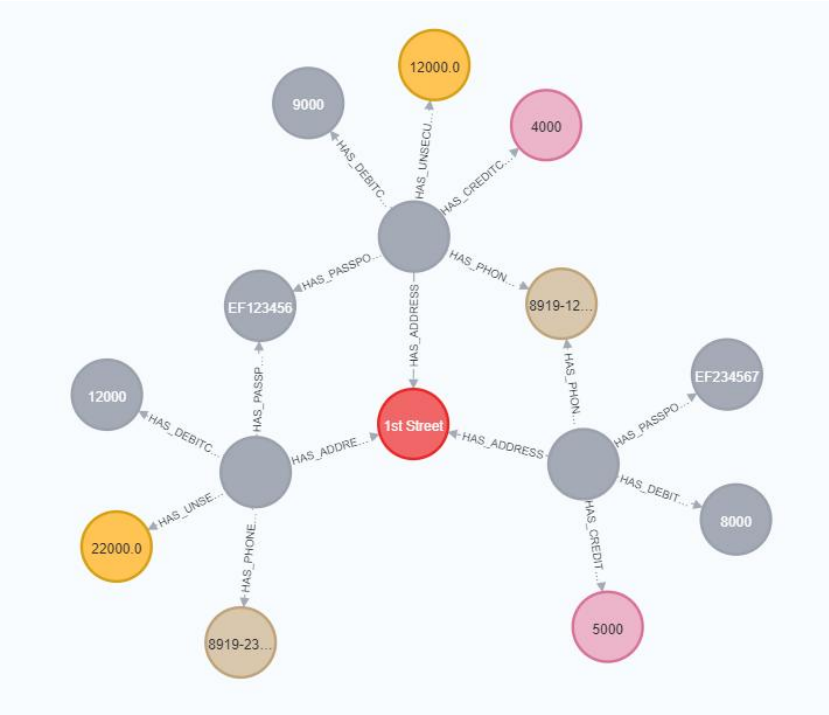
6. Implement

Query 1

```
// Create account
CREATE (account1:Account {
FirstName: "Jack",
LastName: "Smith",
FullName: "JackSmith" })

CREATE (account2:Account {
FirstName: "Dan",
LastName: "Brown",
FullName: "DanBrown" })
```

```
CREATE (account3:Account {
FirstName: "Marry",
LastName: "Williams",
FullName: "MarryWilliams" })
// Create Address
CREATE (address1:Address {
Street: "1st Street",
City: "Moscow",
ZipCode: "101111" })

// Connect 3 account holders to 1 address
CREATE (account1)-[:HAS_ADDRESS]->(address1),
(account2)-[:HAS_ADDRESS]->(address1),
(account3)-[:HAS_ADDRESS]->(address1)

// Create Phone Number
CREATE (phoneNumber1:PhoneNumber { PhoneNumber: "8919-123-
456" })

// Connect 2 account holders to 1 phone number
CREATE (account1)-[:HAS_PHONENO]->(phoneNumber1),
(account2)-[:HAS_PHONENO]->(phoneNumber1)

// Create PassportNo1
CREATE (PassportNo1:PassportNo { PassportNo: "EF123456" })

// Connect 2 account holders to 1 PassportNo1
CREATE (account2)-[:HAS_PASSPORTNO]->(PassportNo1),
(account3)-[:HAS_PASSPORTNO]->(PassportNo1)

// Create PassportNo2 and connect account1
CREATE (PassportNo2:PassportNo { PassportNo: "EF234567" })<-
[:HAS_PASSPORTNO]-(account1)

// Create Debit Card and connect account1
CREATE (DebitCard1:DebitCard {
AccountNumber: "011111111",
Balance: 8000 })<-[:HAS_DEBITCARD]-(account1)

// Create Credit Card and connect account1
CREATE (CreditCard1:CreditCard {
```

```
AccountNumber: "111111111",
Limit: 5000, Balance: 2000,
ExpirationDate: "01-01",
SecurityCode: "123" })<-[:HAS_CREDITCARD]-(account1)

// Create Debit Card and connect account2
CREATE (DeditCard2:DeditCard {
AccountNumber: "022222222",
Balance: 9000 })<-[:HAS_DEBITCARD]-(account2)

// Create Credit Card and connect account2
CREATE (CreditCard2:CreditCard {
AccountNumber: "122222222",
Limit: 4000, Balance: 1500,
ExpirationDate: "01-02",
SecurityCode: "456" })<-[:HAS_CREDITCARD]-(account2)

// Create Unsecured Loan and connect account2
CREATE (UnsecuredLoan1:UnsecuredLoan {
AccountNumber: "211111111",
Balance: 9000,
APR: .0541,
LoanAmount: 12000.00 })<-[:HAS_UNSECUREDLOAN]-(account2)

// Create Debit Card and connect account3
CREATE (DeditCard3:DeditCard {
AccountNumber: "033333333",
Balance: 12000 })<-[:HAS_DEBITCARD]-(account3)

// Create Unsecured Loan and connect account3
CREATE (unsecuredLoan2:UnsecuredLoan {
AccountNumber: "222222222",
Balance: 16341.95, APR: .0341,
LoanAmount: 22000.00 })<-[:HAS_UNSECUREDLOAN]-(account3)

// Create Phone Number and connect account3
CREATE (phoneNumber2:PhoneNumber {
PhoneNumber: "8919-234-567" })<-[:HAS_PHONENO]-(account3)

RETURN *
```



Find account holders who share more than one piece of legitimate contact information.

Query 2

```
MATCH (account:Account)-[]->(contactInformation)
WITH contactInformation,
count(account) AS RingSize
MATCH (contactInformation)<-[]-(account)
WITH collect(account.FullName) AS AccountHolders,
contactInformation, RingSize
WHERE RingSize > 1
RETURN AccountHolders AS FraudRing,
labels(contactInformation) AS ContactType,
RingSize
ORDER BY RingSize DESC
```

	FraudRing	ContactType	RingSize
1	["MarryWilliams", "DanBrown", "JackSmith"]	["Address"]	3
2	["DanBrown", "JackSmith"]	["PhoneNumber"]	2
3	["MarryWilliams", "DanBrown"]	["PassportNo"]	2

Determine the financial risk of a possible fraud ring.

Query 3

```
MATCH (account:Account)-[]->(contactInformation)
WITH contactInformation,
count(account) AS RingSize
MATCH (contactInformation)<-[]-(account),
(account)-[r:HAS_CREDITCARD|HAS_UNSECUREDLOAN]-
>(unsecuredAccount)
WITH collect(DISTINCT account.FullName) AS AccountHolders,
contactInformation, RingSize,
SUM(CASE type(r)
WHEN 'HAS_CREDITCARD' THEN unsecuredAccount.LIMIT
WHEN 'HAS_UNSECUREDLOAN' THEN unsecuredAccount.Balance
ELSE 0
END) AS FinancialRisk
WHERE RingSize > 1
RETURN AccountHolders AS FraudRing,
labels(contactInformation) AS ContactType,
RingSize,
round(FinancialRisk) AS FinancialRisk
ORDER BY FinancialRisk DESC
```

	FraudRing	ContactType	RingSize	FinancialRisk
1	["MarryWilliams", "DanBrown", "JackSmith"]	["Address"]	3	25342
2	["MarryWilliams", "DanBrown"]	["PassportNo"]	2	25342
3	["DanBrown", "JackSmith"]	["PhoneNumber"]	2	9000

Conclusion

As the result, we can see the accounts of those public information, and you can also check the possible financial risks of the fraud circle and predict the composition of the members in the fraud circle.

Bibliography

1. Fraud Examination. 6 Edition. / W. Steve Albrecht, Mark F. Zimbelman, Chad O. Albrecht, Conan C. Albrecht. - Published 2018 by Cengage Learning International Edition, 696 pp.
2. AI and Machine Learning: Harness the world's leading AI-powered fraud detection platform. [Электронный ресурс] : Fraud.net – Режим доступа: https://fraud.net/solutions/ai-machine-learning/
3. Arianit Mehana, Krenare Prieva Nuci. Fraud detection using data-driven approach. - arXiv:2009.06365v1 [cs.LG] 8 Sep 2020 7pp [Электронный ресурс]: ArXiv.org – Режим доступа :https://arxiv.org/pdf/2009.06365.pdf
4. Michael Wallner. Event-based fraud detection with direct customer calls using Amazon Connect. on 14 Jun 2021 in Amazon Connect [Электронный ресурс]: Amazon.com - Режим доступа: https://aws.amazon.com/blogs/machine-learning/event-based-fraud-detection-with-direct-customer-calls-using-amazon-connect/
5. L. Zheng, G. Liu, C. Yan and C. Jiang, "Transaction Fraud Detection Based on Total Order Relation and Behavior Diversity," in IEEE Transactions on Computational Social Systems, vol. 5, no. 3, pp. 796-806, Sept. 2018, doi: 10.1109/TCSS.2018.2856910.
6. Stop Fraud Rings in Their Tracks with Neo4j. [Электронный ресурс]: Use Cases: Fraud Detection & Analytics — Режим доступа: https://neo4j.com/use-cases/fraud-detection/
7. Everett Griffiths. Using Docker in API Gateway and Microservice Development. [Электронный ресурс]: CloudBees – Режим доступа: https://www.cloudbees.com/blog/using-docker-in-api-gateway-and-microservice-development
8. Tahereh Pourhabibi, Kok-Leong Ong, Booi H. Kam, Yee Ling Boo, Fraud detection: A systematic literature review of graph-based anomaly detection approaches, Decision Support Systems, Volume 133, 2020, 113303, ISSN 0167-9236,[Электронный ресурс]: Science Direct – Режим доступа: https://doi.org/10.1016/j.dss.2020.113303
9. Clive Edwards. Using Docker and Kubernetes to produce a scalable fraud detection API. [Электронный ресурс]: cedwards.info - Режим доступа: https://www.cedwards.info/projects/fraud/
10. Polong Lin, Pavan Kattamuri. How to build a serverless real-time credit card fraud detection solution. [Электронный ресурс]: Google Cloud –

Режим доступа: https://cloud.google.com/blog/products/data-analytics/how-to-build-a-fraud-detection-solution

11. Build Real-Time Fraud Detection and Analytics with Confluent. [Электронный ресурс]: confluent.io – Режим доступа: https://www.confluent.io/use-case/fraud-detection/